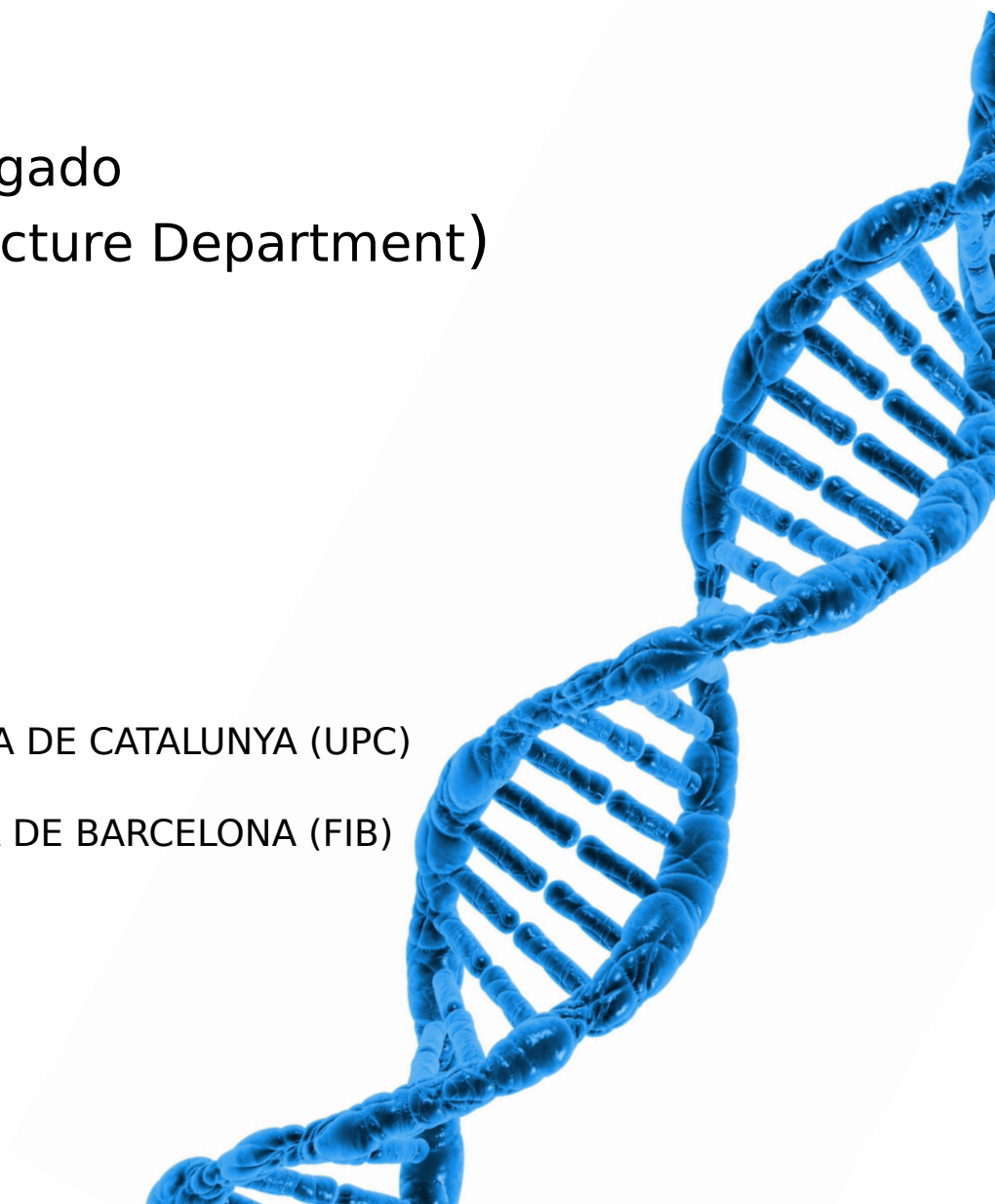
CRITICAL ANALYSIS AND COMPARISON OF DATA PROTECTION TECHNIQUES FOR GENOMIC DATA SETS

Master thesis
by Daniel Naro

Master in Innovation and Research in Informatics
(Computer Networks and Distributed Systems)

04/07/2016

Advisor: Jaime Delgado
(Computer Architecture Department)



UNIVERSITAT POLITÈCNICA DE CATALUNYA (UPC)
– BarcelonaTech
FACULTAT D'INFORMÀTICA DE BARCELONA (FIB)

CRITICAL ANALYSIS AND COMPARISON OF DATA PROTECTION TECHNIQUES FOR GENOMIC DATA SETS

Daniel NARO

MASTER THESIS

DIRECTOR
Jaime DELGADO
Computer Architecture Department

Contents

Abstract

This work aims at contributing to the definition of a new file format for genomic data. In the last years we have seen improvements in our ability to sequence the DNA, which means that the amount of genomic data increases at a growing pace. This creates a need for a new file format specifically intended for the compression of such information. Genomic data is extremely sensitive by nature: it has intrinsic properties for individual identification, ancestry discovery, and disease prediction among others.

This motivates the definition of security strategies for the new file. The present work focuses on contributing to this task. We divide the work in two parts. First we review the research trends. Currently, the main usage of genomic data is through an aggregation of individuals: for medical studies it is helpful to search the common mutations among persons suffering of the same disease and compare the findings to what is observed in a healthy group. This approach was believed to be secure: at no point the data of one individual is disclosed, only statistics over the whole population. However, Homer et al. published a way to infer the presence of one individual in a mixture. This has led to the current orientation of research which examines closely this particular issue. We review in this work what Homer et al. proposed and ways which have been taken to increase the efficacy of this attack, but also the doubts about its realism. Although the last point can be considered an open question, many publications aim at protecting against Homer's attack: how to publish statistical information about the aggregation without endangering the privacy of the individual. A technique to achieve this is differential privacy, which consists in adding noise to the response in order to hide differences between neighbouring sets (i.e. sets which differ by possessing one individual more or less). Although this introduces a trade-off between privacy and utility, and though some claim that no good middle point can be found, this approach has gained momentum, as proved by the different papers we review.

We can also find literature on how to guess portions of the DNA that have been erased for privacy reasons. This attack is of special interest for architectures providing rules on how to access genomic data. For example, an individual could say that just some of his/her genes are to be accessible. This approach is described by different services aiming at providing a more specific control over the data.

The use of genomic information is not bounded only to research purposes. Another goal is to be able to estimate the risk of certain diseases through the analysis of DNA, but also to offer other services like genealogy tools. In order to give access to such applications, the main approach is to envision a cloud platform where the user could accept to let different algorithms run over his records. In order to avoid misuse, the common consensus is that the access to the data has to be regulated through policies. There are multiple ideas, but all rely on a repository

of data offering some API. Requests coming through this API are verified, and only if the request appears legitimate, the result is returned. Another approach which avoids the use of any rule is the path offered by homomorphic cryptography. The idea behind this is that we can pack all the data in an encrypted input. The program executed on this data is unable to decrypt it, however, its output is still meaningful when decrypted. This allows individuals to have a computation executed without revealing any information. This approach is still computationally costly, but there is much research being done on this topic.

The publications we review give many insights in what the believed threats for genomic data are, and in ways to address them. However, we do not find many studies on how to protect the DNA records within the file: the publications take the point of view of a layer above, where a privacy-aware access to the data stored in the file is offered.

In the second part of the present work we try to contribute to topics which are rarely treated in current research. Among other things we make a proposal on how to achieve protection within the file, by describing ways to apply the rules directly to the data, encrypting only certain portions of the file. The idea is that with minimal communication we could grant a new access right to an encrypted region by sending the key for this particular section of the DNA.

We first analyse how to protect a file format of our invention. The invented file format is by no means a proposition for a file specification, it is just an example on which to build an approach for constructing the encryption strategies. Then we apply these strategies to two currently used formats for genomic data, namely the SAM and the CRAM format. SAM is an earlier file type, which attempts at reducing the size of the file, but does not focus on compression (we should note, however, that its binary counterpart BAM does). The CRAM format is far more structured and offers a hierarchical organization of containers which aggregate blocks of data. We see how this structure helps when encrypting some portions of the file.

The CRAM format is one of those achieving better compression scores. Therefore we can expect the future format to take a similar structure. Our findings are therefore likely to be transferable.

We also propose a strategy on how to split the DNA records into multiple files. This procedure allows to gain utility by sharing the information with multiple parties who are at the same time prevented from colluding. This way to proceed should be applicable to any file specification.

Chapter 1

INTRODUCTION

Deoxyribonucleic acid (DNA) was first observed around the middle of the 19th century and since then we have kept deepening our knowledge about it. The genetic material has unique properties, especially when we compare it to other types of medical metrics.

For example an ECG recording helps understand the current physical state of a patient, but it does not have the potential to identify an individual as a dental radiography has. As frequently depicted in popular culture, there are different ways to identify a person: through fingerprints, dental records or, our point of interest, using DNA samples. However, the DNA is more than just a mere tool to identify persons. In [1], the authors establish a list of special features the DNA has:

- Uniqueness: as we have said, the DNA is unique to each individual. Sometimes there are just some tiny variations, as in the case of twins, but in the regular case it is easy to distinguish persons according to their DNA.
- Predictive capability: we are also familiar with the fact that the DNA encodes all our body. Thus, thanks to its study, we can infer the risk of catching a particular disease many years before any symptom appears.
- Immutability: Globally, the genetic code does not change over time. This opposes it clearly to our previous example of an ECG recording where a new healthier lifestyle and/or medication can have huge impact.
- Requirement of testing: according to the authors, genetical diseases need more than the usual clinical tests to be diagnosed. From their point of view, the actual genetical test is frequently required to correctly diagnose the disease.

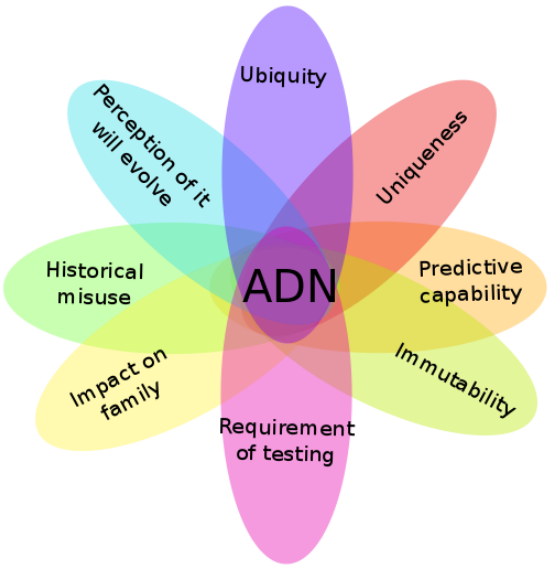


Figure 1.1: Venn diagram representing the special characteristics of DNA

- Historical misuse: we already have examples of people trying to use DNA readings for eugenics intentions. This can only be a word of caution regarding possible abuse.
- Impact on family: as commonly known, blood relatives share part of their DNA, which explains the physical likelihoods among them. The fact that it might be possible to discover something about one individual and be able to extrapolate this finding to his or her family is a new type of privacy threat.
- Evolving perception: not everybody has the same understanding of the mechanics in genetic information, and even amongst researchers we have to expect frequent breakthroughs in the next years. This implies that our perception of the DNA information will evolve as in the past (and currently) our perception of other diseases has changed. (In some cases even, we classified things as diseases that we do not classify as such any more.)
- Ubiquity: according to the authors, we leave so many biological footprints behind us (such as hair and saliva), that DNA can be viewed as ubiquitous.

As we have seen previously, we can find other things which have one or more than one of these features, but DNA is indeed unique in the fact that it combines all of these.

These special features have increased the interest of researchers in using DNA in their studies. As always when there is a new demand, there is soon enough

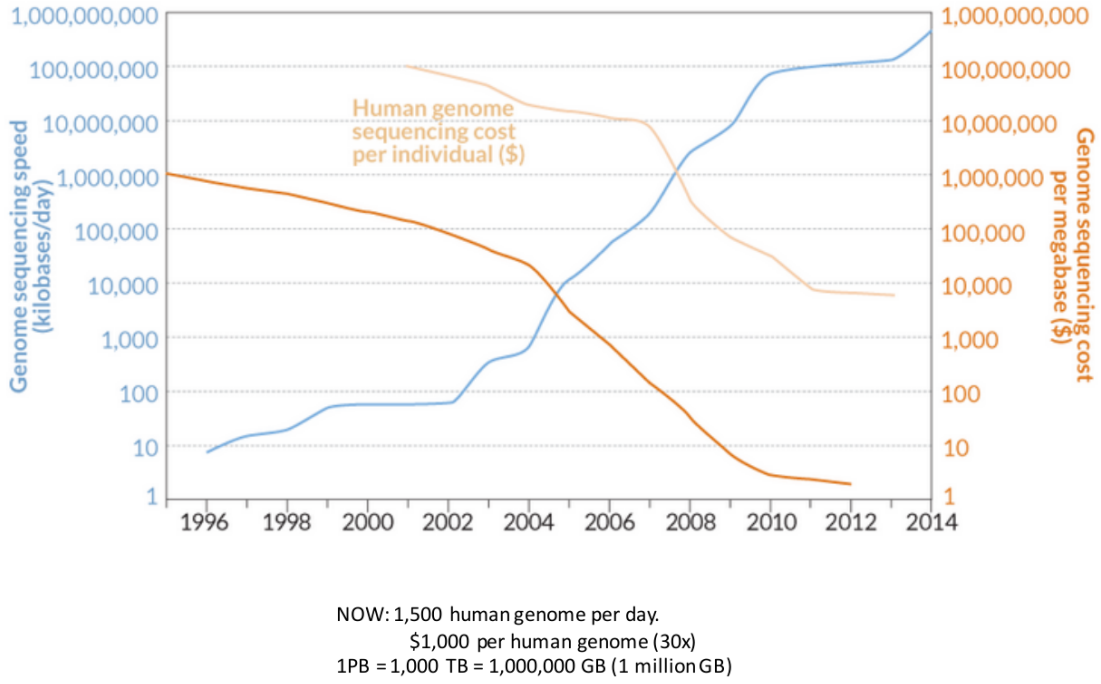


Figure 1.2: Analysis of cost evolution for the genome sequencing, copied from Yong Zhang 's presentation *The Time of Peta-byte is Coming*, January, 23rd 2016

a solution which is found for it. In this case the solution is the development of new hardware which is able to perform the reading of the DNA. This endeavor is already some years old, and we see yet again a very familiar pattern. As can be seen in other fields such as Big Data, physical studies or similar, we have engineered ways to produce huge amounts of information, and, maybe more importantly, we have learned to decrease the cost to obtain it. All of this combined, has created a growth curve which requires to find new solutions at the Information Technology level to cope with it.

The path taken by the DNA analysis is exactly the same (see Figure 1.2), and now we need to find the solutions to keep the pace of the hardware's evolution. Some issues are well-known, for example the requirement in space, which also leads to difficulties in simply sending the information over the Internet, or the need to efficiently compute analysis over this data. This challenges can be faced by the design of specially dedicated compression algorithm, much as we were to able to achieve with previous types of information such as images, audio and video. However due to the specificity of the DNA we face additional challenges.

As we have seen, DNA is immutable over a lifetime, and it is even inherited by one's offspring. DNA has also intrinsic identification capabilities which might

render certain anonymization strategies useless. In order to gain the willingness of individuals to share their DNA for studies, we nevertheless need to address this challenge, and at the same time we must cope with the file weight issue in order to define a file standard to store and make use of genomic information.

The ISO/IEC MPEG group has started a standardization project which aims at solving both issues, weight and security. This group is a subentity of The ISO/IEC JTC 1 which is a joint technical committee of the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) which aims at developing standards for information technology and communications. The subcommittee 29 (SC 29) within ISO/IEC JTC 1 is devoted to the coding of audio, picture, multimedia and hypermedia information. The ISO/IEC JTC 1/SC 29 has in turn formed different working groups, among others WG 11, the working group dedicated to the coding of moving pictures and audio. Its official designation is ISO/IEC JTC 1/SC 29/WG 11, but the working group is commonly known as MPEG. In its past meetings, the group has drawn a list of requirements for both the compression and the security aspect. The objective is to use the experience gathered in compressing other types of files in order to define a new file format for genomic data which will obtain better results than the currently used specifications and take into account the privacy issues within the file itself.

The present work will focus on the strategies that have already been devised to guarantee the security of genomic data, and will then turn to make a proposal on how to apply them to the file formats currently in use. First we will introduce the required basic concepts of the genome, its sequencing, and the workflow to make use of it. Then we will review the literature in search of strategies and solutions to protect such information. This chapter is divided into four parts which correspond to the main trends currently present in the literature. We review Homer's attack, which consists in discovering the presence of individuals in study groups where the participants were not disclosed. We then analyse publications on differential privacy, a field aiming at providing information on a group without endangering the privacy of individuals. We then review studies on another attack which consists in inferring non-disclosed parts of the DNA records of an individual. Finally, the last part is dedicated to the publications concerning homomorphic encryption and its application to genomic data. Homomorphic encryption refers to the field of study where the data is encrypted in such a way that computations are still possible, but the party doing the computation is unable to retrieve any piece of information from it. This research path is still in its early stages, which explains that not all publications we see offer ideas ready for production.

We then need to compare the current proposals to the goals fixed by the ISO/IEC MPEG group for a file format for DNA records. The approach taken by the current publications and the standardization group differ in some aspects, but

they share most of their goals. Finally we will make a step towards a proposal: we first work on an invented file format in order to develop some ideas and then apply them to existing formats.

Chapter 2

THE GENOME

2.1 Introduction to the genome

Our bodies are the result of combining different systems (e.g. the circulatory system, the digestive system,...). Each system is the sum of different organs, which are made up of tissues which are the aggregation of different cells. There are different types of cells, each with a different function. Nevertheless, all cells have the same coding (i.e. building instructions for the different molecules our body produces), the differences among types reside in the "interpretation" of the coding (e.g. only the beta cells in the pancreas will read the portion of DNA which encodes the recipe for insulin).

The cell's coding, or genetic information, is stored in the Deoxyribonucleic acid (DNA)) molecules. These molecules are most neatly distinguishable when the cell is preparing to divide itself: either for *mitosis* [1](producing two identical cells), or *meiosis* (producing two different cells). During these two operations, the DNA molecules take clearly defined structures known as chromosomes. We humans have 23 pairs of chromosomes, each chromosome being made of one DNA molecule. Therefore we have by default 46 molecules of DNA. During mitosis one copy of each DNA molecule is produced, therefore during some periods the cells have doubled the number of DNA molecules.

As we have seen previously, the DNA is one of the most elemental building blocks of our bodies. This gives many reasons to investigate how it works, and how differences among codings can lead to advantages or disadvantages. In order to understand where the differences originate, we first need to give some more details about the structure of the DNA molecules. DNA molecules are made of two *strands*. Each strand is a sequence of *nucleotides*, and there are four types

[1]A glossary of technical terms is enclosed at the end of the study. The number at the end of each entry corresponds to the page where the term is first defined.

of nucleotides: *adenine* (A), *thymine* (T), *guanine* (G), and *cytosine* (C). The nucleotides at a given position of one strand let us know the nucleotide at the same position on the other strand: we refer to the combination of the two nucleotides as a *base pair*, and there are two different such pairs: A-T and G-C. When adding up all 46 molecules of DNA we have around 3 billion base pairs, which are almost identical among all individuals.

In order to simplify the understanding of this structure we have defined the concept of a *gene*. A gene is a certain location of the genetic information which encodes a given protein. As we said, the genetical information is almost the same among all humans: there are just some punctual mutations which can produce differences. These mutations lead to the fact that for one given portion of the genetic information or gene, we have different versions we call *alleles*.

During the formation of the *gametes* or sexual cells, as a result of meiosis, we produce two cells with half the genetic information we possess. Mutations will occur during this process, but the vast majority of the coding will be inherited by the offspring as a combination of each of the half genotype provided by each parents. This explains the physical similarity we see among persons of the same family. The genetic information resulting of the fecundation remains the same during the whole life (except for mutations due to radiation and similar incidents, or a viral infection).

One can classify mutations as either an insertion, a deletion or a modification. When this mutation only affects one nucleotide, we call this *Single-Nucleotide Polymorphism* (SNP). In order to understand the possible effects of one SNP we can turn to the so-called genetic code table (Table 2.1). Proteins are built aggregating specific amino acids in a specific order. In order to know which amino acid comes next, the cell decodes the information stored in the DNA: groups of three nucleotides, called *codon*, encode one amino acid. We can see in Table 2.1 that one change in a letter can already have consequences, despite the fact that there are three nucleotides involved in the encoding. For example, the codon 'GUU' encodes valine (noted as *Val* in the table), but just by changing the first letter to 'A', the result is changed into isoleucine (noted as *Ile* in the table). The most radical change in a protein due to a mutation, however, might occur when a mutation leads to a codon which encodes a 'STOP', i.e. the codon indicating the end of a protein. In other words, just one mutation can shorten the obtained protein.

These mutations are at the origin of some of the differences we observe: some mutations will just affect for example hair colour, but in other cases one mutation can explain a bigger risk of getting a disease: for example, a higher risk of contracting Alzheimer's disease. This fact motivates comparative studies among broad populations of individuals, in order to understand which mutations lead to which diseases. As always these studies need to compare healthy populations and

First letter	Second letter: U			Second letter: C			Second letter: A			Second letter: G			Third letter
	U	UUU	Phe	UCU	Ser		UAU	Tyr		UGU	Cys		U
		UUC		UCC			UAC			UGC			C
		UUA	Leu	UCA			UAA	STOP		UGA	STOP		A
		UUG		UCG			UAG	STOP		UGG	Trp		G
	C	CUU	Leu	CCU	Pro		CAU	His		CGU	Arg		U
		CUC		CCC			CAC			CGC			C
		CUA		CCA			CAA	Gin		CGA			A
		CUG		CCG			CAG			CGG			G
	A	AUU	Ile	ACU	Thr		AAU	Asn		AGU	Ser		U
		AUC		ACC			AAC			AGC			C
		AUA		ACA			AAA	Lys		AGA	Arg		A
		AUG	Met	ACG			AAG			AGG			G
	G	GUU	Val	GCU	Ala		GAU	Asp		GGT	Gly		U
		GUC		GCC			GAC			GGC			C
		GUA		GCA			GAA	Glu		GGA			A
		GUG		GCG			GAG			GGG			G

Table 2.1: The genetic code table. Summarizes which amino acid will be added to the protein being synthesized depending on the codon (sequence of three nucleotides) being read

populations with the disease. This kind of studies is denoted as *Genome Wide Association Studies* (GWAS).

Once a given mutation is understood, we can use this information, for example to evaluate the risk of a given disease for a patient whose genome is known. This usage of the genetic information in order to offer a service to one person is called *Direct-To-Consumer* (DTC). But besides the medical usage there are also more unexpected applications such as tools for genealogy (finding relatives) or cosmetics (compositions which better suit certain skins).

2.2 Retrieving the genome

The genome needs to be retrieved from cells and the results of this operation need to be post-processed. Currently, it is not possible to sequence the whole genome of an individual at a time. In fact, just some chunks of information are retrieved subsequently: they represent one reading of one portion of the genome. The length of these readings depends on the equipment which is used, but in general the longer the read, the more error prone it is. In this early stage of the DNA

analysis workflow we do not know from which region of the genome the reading is, and (some of) the nucleotides which we believe to have seen might in fact be incorrectly identified.

Such reads can be represented in the FASTA format. We first store the id of the read in the first line: we prepend the symbol ">" to establish the nature of such a line, and then we give the sequence of nucleotides. This sequence can span over multiple lines. The end result looks as follows:

```
>HWUSI-EAS100R:6:73:941:1973#0/1
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
```

After iterating these reading operations many times, we reach a point where it is very likely that we have more than one reading for each nucleotide of one area of interest in the genome, or for the whole code. Now, this raw data needs to be processed in order to make it useful. As we have seen in chapter 2.1, the vast majority of the human genome is common to everybody. The post-processing builds upon this fact by comparing each read to a reference genome. It finds the most similar location (i.e. the one with the smallest edit distance), and assumes that that read comes from that location. After this operation, only the differences with the reference genome should be of interest: everything else is supposedly the same as our reference. However, not every discrepancy with the expected genome is actually a mutation. In fact, divergences could be errors of the sequencing. The trend is therefore to store all the raw data, since in some future we might better understand it and be able to better differentiate between mutation and measurement noise. In order to grade the assumable reliability of the measures different approaches exist.

The first one is to add a new possible output to the four nucleotides: alongside the usual A, C, T and G, we add a new symbol N indicating that no decision could be taken for the identification of that given position. In the same spirit, the International Union of Pure and Applied Chemistry (IUPAC) published a nucleic acid notation which includes symbols for the different doubts: on top of A, G, C, T, U (in the case of using it with RNA), it gives symbols which encode either one or the other nucleotide (for example W encodes either A or T, and S either C or G), and also symbols which indicate that it might have been any nucleotide except one (for example B means either C, G, and T ruling out A as an option).

Another strategy to indicate the confidence of a measure is to simply grade the read at that position. To this end we use the FASTQ format: in this format we indicate every nucleotide forming the read and associate a mark in the range from 0 (noted as "!") to 93 ("~") using the order of symbols in the ASCII table. The end result looks as follows (being the counterpart of the previous FASTA example):

```
@HWUSI-EAS100R:6:73:941:1973#0/1
```

```
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
+
!''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65
```

The first line indicates the name of the read (after the @), the second line is the actual read, the third line is becoming obsolete, and the fourth one contains the marks of the read.

As we previously said, once we have the different readings we try to map them on a reference genome: building on top of the fact that most of the DNA is shared among individuals, the reference genome is a kind of average code. We now search for each read which portion of the reference is the most resembling: we expect most mutations to be Single Nucleotide Polymorphism (SNP), therefore there should be plenty of information in each read to find the "right place". With "right place" we refer to the one which minimizes the edit distance, i.e. we need the fewest modifications to go from the reference to the read (the modifications being a nucleotide permutation, deletion or insertion). The end result is something similar to the following, but far longer and with far more reads.

```
Coor    12345678901234  5678901234567890123456789012345
ref     AGCATGTTAGATAA**GATAGCTGTGCTAGTAGGCAGTCAGCGCCAT

+r001/1       TTAGATAAAGGATA*CTG
+r002        aaaAGATAA*GGATA
+r003      gcctaAGCTAA
+r004                     ATAGCT..............TCAGC
-r003                            ttagctTAGGC
-r001/2                                       CAGCGGCAT
```

This alignment can then be stored to memory using different formats, for example the SAM format. The previous alignment in SAM format is the following one:

```
@HD VN:1.5 SO:coordinate
@SQ SN:ref LN:45
r001  163 ref  7 30 8M2I4M1D3M = 37  39 TTAGATAAAGGATACTG *
r002    0 ref  9 30 3S6M1P1I4M *  0   0 AAAAGATAAAGGATA   *
r003    0 ref  9 30 5S6M       *  0   0 GCCTAAGCTAA       * SA:Z:ref,29,-,6H5M,17,0;
r004    0 ref 16 30 6M14N5M    *  0   0 ATAGCTTCAGC       *
r003 2064 ref 29 17 6H5M       *  0   0 TAGGC            * SA:Z:ref,9,+,5S6M,30,1;
r001   83 ref 37 30 9M         =  7 -39 CAGCGGCAT         * NM:i:1
```

Some similarities are clear: for example the first column is the same read identifier, the tenth column is the actual read sequence. On top of that we have extra information such as bitwise flags in column two, the possibility to add quality

information in the eleventh column using the same notation as in FASTQ (in this example this information is omitted using the symbol '*'), or marking the quality of the alignment in column 5. What might be surprising at first is how the indication of the position is done: if we look at our example we have theoretically no read starting at position 9; however, in column 4 it is what we indicate for the reads r002 and r003. In fact, we do not indicate the first position of the read, but rather the first nucleotide of the read which is a match. It is also interesting to note that the second r003 is located at 29: we are therefore using the indexation of the reference, not a special indexation resulting of the different insertions and deletions. For example, in read r001 we detect two nucleotides more than expected after position 14 in the reference, and we just transcribe this as two insertions ("2I") in the 6th column.

The SAM format relying heavily on ACII characters might be too heavy. In order to introduce a first level of weight reduction, the BAM format was introduced. This format is a gzip compression of SAM, but splitting this file in blocks in order to simplify random access.

Alongside SAM/BAM there is also CRAM, which has better compression rates than CRAM in the benchmarks. The overall structure of CRAM files, as described in the specifications for its third version, can be seen in Figure 2.1. As in BAM, compression is done in a per-block way, but there is a differentiation depending on the type. Core data blocks are compressed with bit encoding, external data blocks with byte encoding (external blocks are meant to refer to other blocks using an id).

As in SAM, CRAM has a field to indicate which reference was used. In CRAM, slices have the field for an identification of which reference(s) DNA was/were used. This information is stored in the slice header block. Among other information the header also contains where the alignment of that slice begins, and for how many nucleotides it spans. In order to simplify the query of specific regions of the information, one can generate an index. Cram accepts BAM indexes, but also CRAM indexes: for every slice in the document,this file lists - among other information - which region of the DNA it covers (first position and length) and where the slice can be found in the overall structure of the file.

The core data blocks are the actual collections of CRAM records. CRAM records have similar fields to what we have previously seen: a link to the reference which was used (one of those specified for that slice), read length, the alignment start position, quality scores among others.

We have seen by now the pipeline used from the point where the DNA is sequenced, aligned and written to files. However this is not the end of the pipeline which is shown in Figure 2.2. The next step in the analysis is to detect the variations which are present in the individual (according to the reference genome).

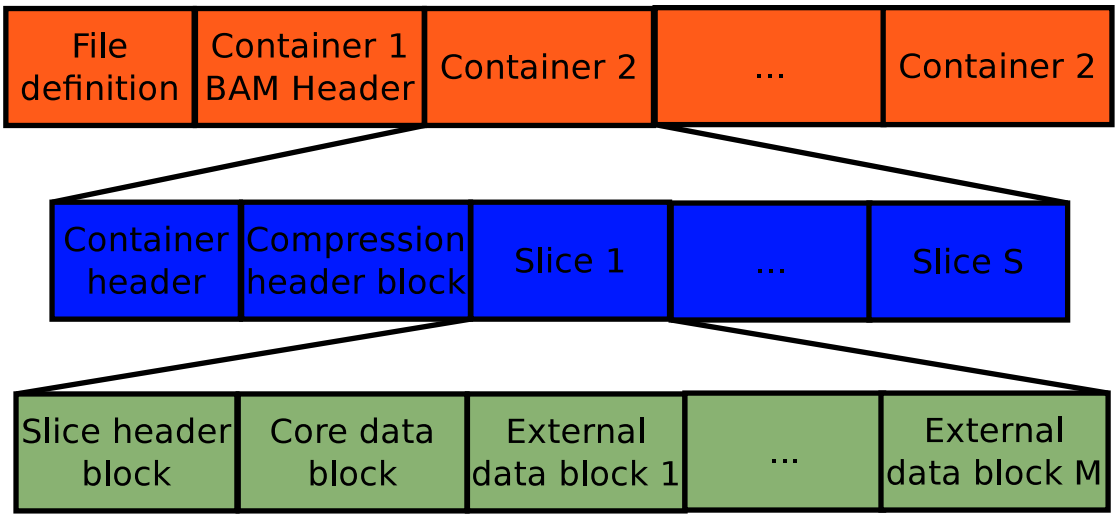


Figure 2.1: CRAM file structure

As always when performing some kind of measurement we will experience measurement noise, i.e. the result of the measure will not be the exact value. As we saw with the FASTA and FASTQ such types of measurement noise are already contemplated in the sequencing of the DNA.

On top of these technical difficulties, there are also intrinsic challenges with DNA. One cell might have been exposed to environmental factors such as radiation, affecting the DNA, while others remained protected from it. In that case two cells of the same individual might have different codings. On a similar note, a virus infection could lead to the fact that some cells have genetic information within the human DNA strands.

However in some cases we will have a consistent set of reads which have sequenced a different nucleotide for one position in the genome. Such a consistency is an indication that it is very likely that in fact the sequenced person presents a mutation at this location. We refer to this step with variation detection, and we summarize the findings in one Variant Code Format (VCF). One example is shown in the File snippet 1): from the lines 23 to 40, we can see how the mutations are listed, indicating for each chromosome and each position what did the reference indicate and what is the actually belief of the value at that position.

Once the mutations are detected and annotated, the genomes of different individuals can be compared in order to find similar patterns among healthy and unhealthy populations. One of the usual patterns to search for is the Minor Allele Frequency (MAF). As we have seen there is a common nucleotide for one given position in the DNA, but if one mutation causes a disease then we will see consistently a minority of individuals having another nucleotide at that location. This is

by definition an allele which is less frequently present. In order to discover which mutations are at the roots of a disease, it is therefore useful to compare such Minimum Allele Frequency for healthy and unhealthy populations in order to construct hypothesis to explain the observed phenotype and maybe finding ultimately a new diagnose tool or even a cure.

It is interesting to note that in the papers published on security for genomic data, so many work on this last stages of the pipeline. The main focus is to provide secure tools to retrieve securely the MAFs, and perform other statistical methods for the analysis of genomic data. Since they make an important abstraction, forgetting about the actual process of sequencing, aligning and detecting variants, many papers take a simplified notations for the mutations. Namely just assigning a numerical score to the (non)mutation at one particular location. The idea is that, as we have two chromosomes (one from the mother's side and one from the father's side), we can express the mutations with a three value scale. For example, 0 can be the mark for no mutations, 1 for a mutation on both chromosomes and then 0.5 indicates the case of just a mutation on one of the chromosomes. This notation simplifies indeed the expression of queries, but it also hides many interesting findings: namely those which are not a SNP mutation and which are therefore not easy to query using just a position on the reference genome.

```
##fileformat=VCFv4.0
##fileDate=20110705
##reference=1000GenomesPilot-NCBI37
##phasing=partial
##INFO=<ID=NS,Number=1,Type=Integer,Description="Number of Samples With Data">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Total Depth">
##INFO=<ID=AF,Number=.,Type=Float,Description="Allele Frequency">
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=DB,Number=0,Type=Flag,Description="dbSNP membership, build 129">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##FILTER=<ID=q10,Description="Quality below 10">
##FILTER=<ID=s50,Description="Less than 50% of samples have data">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##FORMAT=<ID=HQ,Number=2,Type=Integer,Description="Haplotype Quality">
#CHROM POS    ID        REF  ALT     QUAL FILTER INFO                              FORMAT      Sample1        Sample2        Sample3
2      4370   rs6057    G    A       29   .      NS=2;DP=13;AF=0.5;DB;H2           GT:GQ:DP:HQ 0|0:48:1:52,51 1|0:48:8:51,51 1/1:43:5:.,.
2      7330   .         T    A       3    q10    NS=5;DP=12;AF=0.017               GT:GQ:DP:HQ 0|0:46:3:58,50 0|1:3:5:65,3   0/0:41:3
2      110696 rs6055    A    G,T     67   PASS   NS=2;DP=10;AF=0.333,0.667;AA=T;DB GT:GQ:DP:HQ 1|2:21:6:23,27 2|1:2:0:18,2   2/2:35:4
2      130237 .         T    .       47   .      NS=2;DP=16;AA=T                   GT:GQ:DP:HQ 0|0:54:7:56,60 0|0:48:4:56,51 0/0:61:2
2      134567 microsat1 GTCT G,GTACT 50   PASS   NS=2;DP=9;AA=G                    GT:GQ:DP    0/1:35:4       0/2:17:2       1/1:40:3
chr1    45796269        .       G       C
chr1    45797505        .       C       G
chr1    45798555        .       T       C
chr1    45798901        .       C       T
chr1    45805566        .       G       C
chr2    47703379        .       C       T
chr2    48010488        .       G       A
chr2    48030838        .       A       T
chr2    48032875        .       CTAT    -
chr2    48032937        .       T       C
chr2    48033273        .       TTTTTGTTTTAATTCCT       -
chr2    48033551        .       C       G
chr2    48033910        .       A       T
chr2    215632048       .       G       T
chr2    215632125       .       TT      -
chr2    215632155       .       T       C
chr2    215632192       .       G       A
chr2    215632255       .       CA      TG
```

File Snippet 1: VCF example file

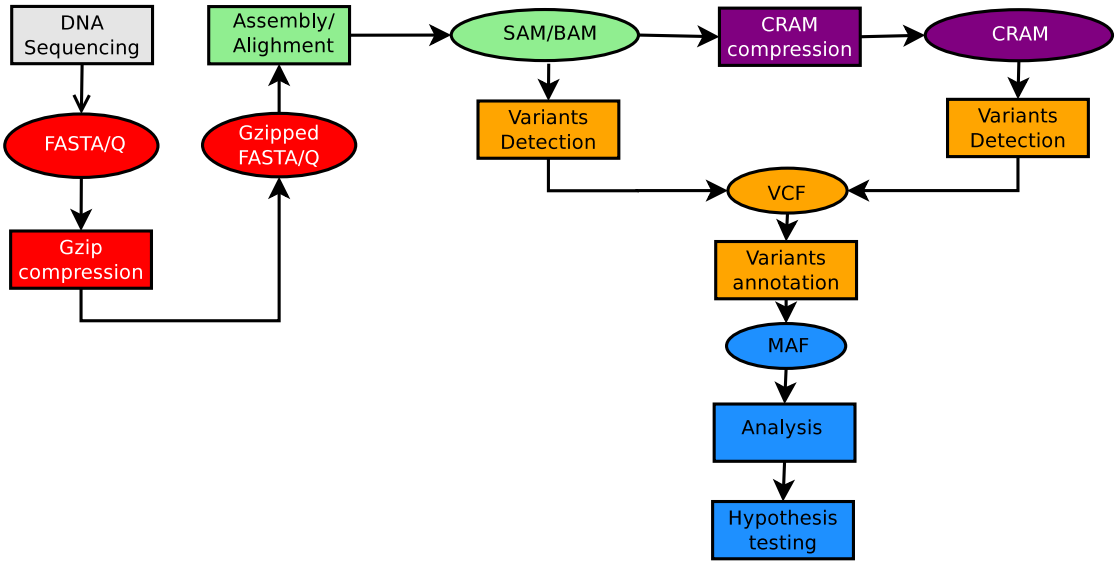


Figure 2.2: Workflow of DNA analysis

2.3 Foreseen usages

As said in Chapter 2.1, we can subdivide the usages of the DNA in two main types: the use in a Genome Wide Association Study (GWAS) and Direct To Consumer (DTC) services. During the "Genomics and Patient Privacy: Research and Practice" workshop at Stanford ([2]), this clear separation in two fields was not present during the presentations. The speakers rather proposed solutions which integrate both views. Michael Snyder for example showed that obtaining very complete data sets periodically has advantages for both fields: as research material, but also in order to detect diseases early and treat them more successfully in a DTC approach.

This same idea of enabling multiple usages is the main motivation for the different app-like usage experiences described. Carl Gunter's presentation ([2]) introduces the use of a cloud repository of genetic data where apps could be run. The apps could be devised for either medical research purposes or direct to consumer services. In contrast to current practice, with such a platform individuals should not send their DNA sequencing to the companies (e.g. www.genepartner.com/, one example given during the workshop) offering the service, but rather accept the execution of the algorithm over their data. In order to achieve this, it is mandatory to define a standard for the data, but also how the API would work and how to ensure reliability and accuracy.

On a similar note, Dave Maher ([2]) from Intertrust Corp presented their cloud platform which allows different services to access the data. Their architecture

already takes into account account individual preferences. According to his presentation, their cloud system contains the encrypted data, and a policy engine which controls the access to the data. The different applications are executed in a trusted environment, and are only granted access through the policy engine. Intertrust takes clearly the route of bringing the computation to the data, arguing that every time copies of the information are sent, its governance is threatened if not lost.

This concern about respecting the will of the patient (or sequenced person) is the central point of Robert Shelton's presentation ([2]). He argues that in order to give incentives to individuals to use and share their genetic information, they need to see that their opinion is taken into account. Basing himself on published statistics, he shows that asking the individuals whether they want to participate or not in a study ensures more engagement. Private access, the platform that Robert Shelton is describing, is based on rules where the patient defines for each gene either whether the data should be publicly available or not, or whether his permission should be asked for. Such a platform offers great potential to simplify the task of the researchers who know that the data they obtain is already compliant with the different laws and the will of the individuals concerned: they will not receive information which is not intended for them.

We should note for completeness that the ISO group considers the use of a standard for genomic data also for forensic and animal genomic sequencing, use cases which are neither a GWAS nor a DTC application.

Chapter 3

CURRENT SECURITY CHALLENGES AND PROPOSED SOLUTIONS

Not so long ago we did not even know how to sequence the genome. It is true that the cost of retrieving the genomic code has now started to decrease drastically, but it is still too soon to have very large collections of data leading to high incentives to attack them. However, due to the utility of sequencing and the fact that it becomes more affordable, it seems reasonable to expect that, in a near future, we will have such big repositories with high incentives.

We are continuously deepening our knowledge about the DNA, genes and mutations. We do not yet know every function of every gene, but we are continuously discovering new applications. Some of them will imply significant issues for privacy. For example, in [3] a method to derive facial composites from 24 SNPs and ancestry information is proposed. This is just one of the examples of how a possibly non-legitimate attack on a DNA repository could be a severe threat to privacy.

As Bradley Malin says in his presentation at Standford's GAPP conference ([2]), we do not have examples of real attacks. What we have are attack experiments run by researchers which are now motivating current research paths. In this chapter we will review these trends.

First, we look into Homer's attack: this attack describes how to infer whether a person is present in a genomic data base, even though we do not have metrics on the individual scale. This attack has motivated many publications in recent years, an important part of which belongs to the field of differential privacy: in this line of work, researchers try to find new paths to publish statistics over a whole population without leaking any information about any individual or sub-group. We group these publications in a section apart, since the attention given to this

domain of study is considerable. Homer's attack is not the only one we review, we also look at publications concerning the attempts to infer portions of DNA which have been censured, as was done with Dr. Watson's records. Finally we see that there is an important trend in genomic computation towards homomorphic encryption, which has promising features when it comes to combining both utility and privacy. But first of all, we need to introduce the concept of *beacon* which is used in many approaches.

Currently one of the main reasons to sequence a person's DNA is the conduction of a GWAS on a specific disease: we know that a particular person has the disease and we want to study his/her DNA along with that of other patients to detect common mutations. In such a case, the research team gathers many sequences which might also be relevant for other studies (e.g. using that DNA as control for another GWAS). The research group would also gather genomic data of healthy individuals as a comparison tool. In order to share the information without putting individuals' privacy at risk, the data is not accessible on a per-individual basis. Instead, the queries which will be accepted concern the whole aggregation. Such queries might refer to a given position in the genome, for example: Is there at least one individual with a mutation at that position? or: What is the proportion of the Minor Allele Frequence (MAF) for this position? Such a repository of data is called beacons. We also have to point out that other repositories exist where one can retrieve entire individual genome records for those studies requiring it. For example the 1000 Genomes Project has sequenced many individuals with different ancestries in order to offer a broad spectrum of genetic material to conduct research on.

3.1 Homer's attack

Homer et al. described in [4] a technique for detecting the presence of an individual in a group. In their publication, they consider the case of forensic analysis, but they also prove the validity of their contribution with DNA sequences from the HapMap repository. They have access to the allele frequencies estimates for the group, and to reference values from a reference population, for each SNP. In order to determine whether the individual is in the mixture they compare whether his or her frequencies are more similar to the reference population or to the mixture. With a statistical test they then give an answer to the question whether the individual is in the mixture or not.

Due to the fact that the frequency of SNPs might be bounded to ancestry, the authors propose solutions to reduce this influence. One of them consists in using SNPs which are known to be less tied to ancestry. The other solution is to use a reference population adapted to the individual who is being searched. By

knowing, or discovering through SNPs, which reference is better suited, we are better equipped to correctly assert whether the individual was in the mixture or not. Since blood-relatives share at least some DNA, this attack can be reformulated to detect someone close to the studied individual in the mixture.

This technique is considered as privacy threatening since being in a study mixture might be an indication of a disease or similar. For example, if it is a study on how some mutations might imply higher threats of becoming addicted, being in the case group mixture of DNAs is something which should remain secret. However, a straightforward implementation of a beacon where every frequency request is answered without any control enables such an attack. This led to some drastic changes in how published genomic data was perceived: the National Institute of Health (NIH) withdrew the data they had published and released a communicate apologizing for the privacy issues they had caused. The NIH was not the only repository to take this solution to the problem posed by Homer et al.

The publication by Homer et al. led to further improvements on the strategy. For example in [5], a mathematical reformulation of the attack was proposed, which also allowed to include prior knowledge to further enhance the reliability.

In [6], Jacobs et al. propose to use a likelihood test to improve Homer's approach: they build upon the logarithm of genotypes in stead of using allele frequencies as Homer does. Their conclusion is that by doing so, they increase the sensitivity of the attack.

Due to various factors, the probability of finding a certain mutation is possibly not independent of finding another. Jacobs et al. stated in 2009 ([6]) that at that point, for their method to behave correctly, they needed to be in linkage equilibrium. In other words, the mutations they were using had to present uncorrelations until the underlying dynamics were better understood. Wang at al. prove in [7] that linkage disequilibrium is in fact a powerful tool to further improve re-identification attacks in Genome Wide Association Studies. According to the authors, just with the results which are usually published after such a study it could be possible to discover the presence of one individual in the case group. Wang et al. describe the likelihood of such an attack as "even more realistic than expected".

However, it is not everyone's opinion that Homer's attack and derivatives are realistic. These methods need input which is not trivially obtained. On top of access to the beacon, or otherwise the statistics of the study, one needs the DNA of the victim and a reference population. Wang et al. in [7] propose to use the HapMap collection of genomes, which are classified according to origins, as reference, but the problem of the victim's DNA is still present. According to Bradley Malin in his Stanford talk ([2]), a group of researchers were asked to evaluate the feasibility of such an attack. In order to do so they used "anonymized" genome

sequences that were available to them, and applied re-identification techniques on aggregations of data. They were able to correctly detect the presence of some individuals in groups, however they were not able to tell who they were. In other words, the DNA has the property to identify uniquely one individual, but you actually need the comparison to assert the match. In their case that meant they knew that the person with that DNA was in the study but without knowing who s/he was, the negative effect of such a breach was greatly limited.

Braun et al. made a follow-up study on Homer's publication: in [8] they show that Homer attack has good sensitivity indeed, but that it lacks in specificity. Some assumptions for the null case in [4] are not met, thus the observed behavior is not the one Homer et al. anticipated. The end result is that true-positives are very likely, but false-positives are also too likely, which undermines the effectiveness of Homer's method. According to Braun et al., the strategy described in the attack is not promising as an attack, but could become a useful statistical tool for some GWAS, if it is modified and further improved,

Bradley Malin ([2]) considers that the current ways of obtaining data are already dissuasive: too many controls such as need to give contact information and even review boards in order to have access to the input for the previously seen family of attacks. From his point of view it takes little more to remove any 'rational' incentive for an attack. By 'rational', the presenter refers to those attacking in order to obtain some profit other than knowing that the attack was successful. He compares estimates of the reward obtained when breaching the security and obtaining a reidentification, to the cost of obtaining the data to perpetrate such an attack. The latter is much easier to evaluate. For the first, he uses estimations of the prejudice in other attacks against medical privacy. Through game theory he proves that it is possible to find a result where no rational person would attempt such action, but where there are still incentives to give access to the data. From his presentation we could conclude that making the process of obtaining the data so costly in time and money is already a guarantee of security, but as he says, it is still hard to evaluate the actual reward of such an attack and therefore it is difficult to know what the cost should be.

The attack published by Homer at al. is based on frequencies. In 2015, Shringarpure et al. introduced a variation on it in [9], which is based on answers to yes or no questions. The idea is that certain beacons allow to query whether, for a given position, they have a certain mutation in the aggregation, whereas the question was previously about the minimum allele frequency for that position (i.e. what is the probability to see the least common variation). We can see the proposal of Shringarpure et al. as an adaptation of a Bloom filter to Homer's attack.

Bloom filters are a tool to detect the presence of one element in a set and

are frequently used in network monitoring. They are specially designed to allow fast insertions and fast tests of presence in the set (the corresponding response has a certain probability of false positives). The idea is that each element which is inserted is hashed with k different hash functions. A bit array is maintained, where each position is related to one of the values of the hash function. When the element is inserted, all bits corresponding to the returned hash values are set to true. A query is then similar to an insertion: we hash the element, but instead of setting all the referenced bits to true, we test whether all of them are true. If not, the element was never introduce; if yes, we return that indeed it might have been introduced, but that there is a false-positive risk. Of course, the higher the value of k, the less likely the false positive is.

As in a Bloom filter, what Shringarpure et al. propose considers that each insertion leaves a certain amount of traces in the aggregation. In this case, instead of having k such traces, we have the amount of SNP mutations of the DNA inserted. If the beacon accepts queries like "Do you have at least one record with a mutation in this position?", then we can readily adapt the regular query in a Bloom filter to this situation based on a succession of such queries. This procedure is summarized in Figure 3.1.

3.2 Differential privacy

One key element to succeed in a Homer-like attack, with high confidence in the results, is to have accurate measurements of the frequencies. When much time and effort have been spent to obtain statistically significant results, it seems odd to add noise to the result, even though it would defend against such attacks. Nevertheless, this is the path which is taken by researchers studying differential privacy ([10]). The idea behind this is to add so much noise to the data that the effects of one individual on the aggregated result is not perceivable any more, but enough accuracy is preserved to draw meaningful conclusions about the the population as a whole. As Dwork ([2]) says: being able to discover that humans have one left foot and one right foot is no privacy challenge. The privacy challenge resides in the fact that we might be able to tell for one specific individual in the aggregation that s/he does have twice the same feet.

In order to achieve differential privacy we add noise to the output. We want to achieve a situation where the result of a query over the aggregated data is virtually the same as the result of the same query over the same data with one individual more or less. We achieve this by adding noise to the result obtained from the aggregated data.

We define a parameter ϵ which gives a sense of the privacy we are striving for. For two neighboring sets, the results of the aggregation function should not differ

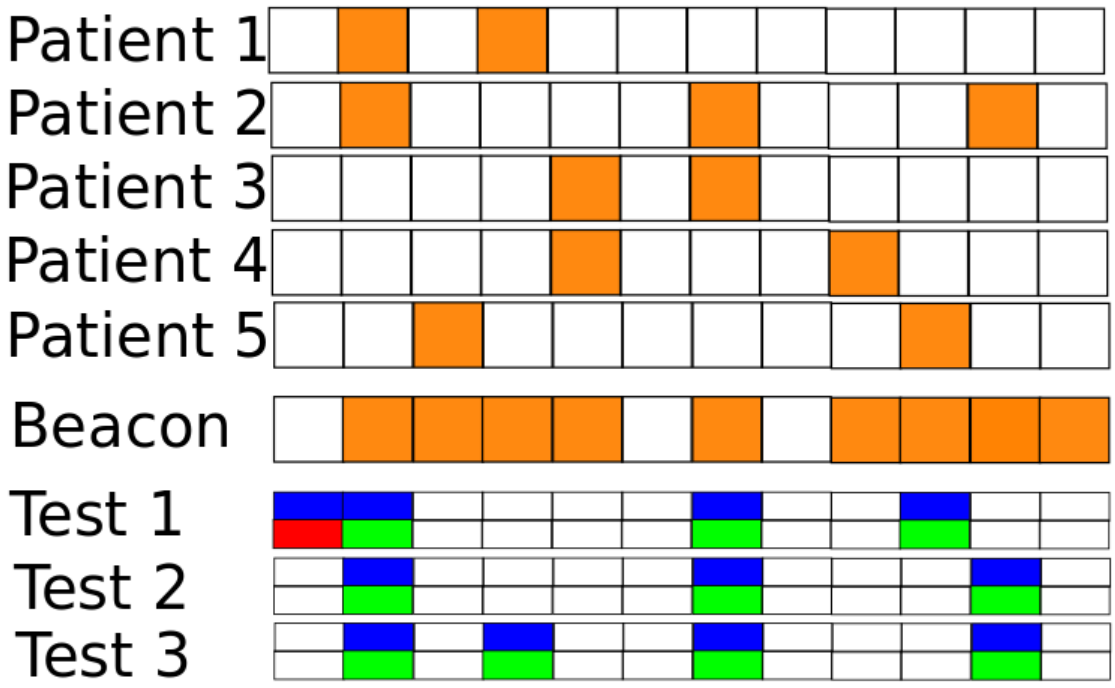


Figure 3.1: Shringarpure et al attack. In the first part we represent the different individuals included in the beacon (we mark in orange the regions where they have a mutation according to the reference). In the second we indicate the state of the beacon: an area is marked as orange (mutated) if for at least one patient that region was not as the reference. We present three test attacks: the result of the first one is negative since a mutation in the first region is unknown; the result of the second one is positive, as we have correctly asserted the presence of patient 2; the result of the third one gives a false positive: we think there is someone with this genomic code, but this is a mistake.

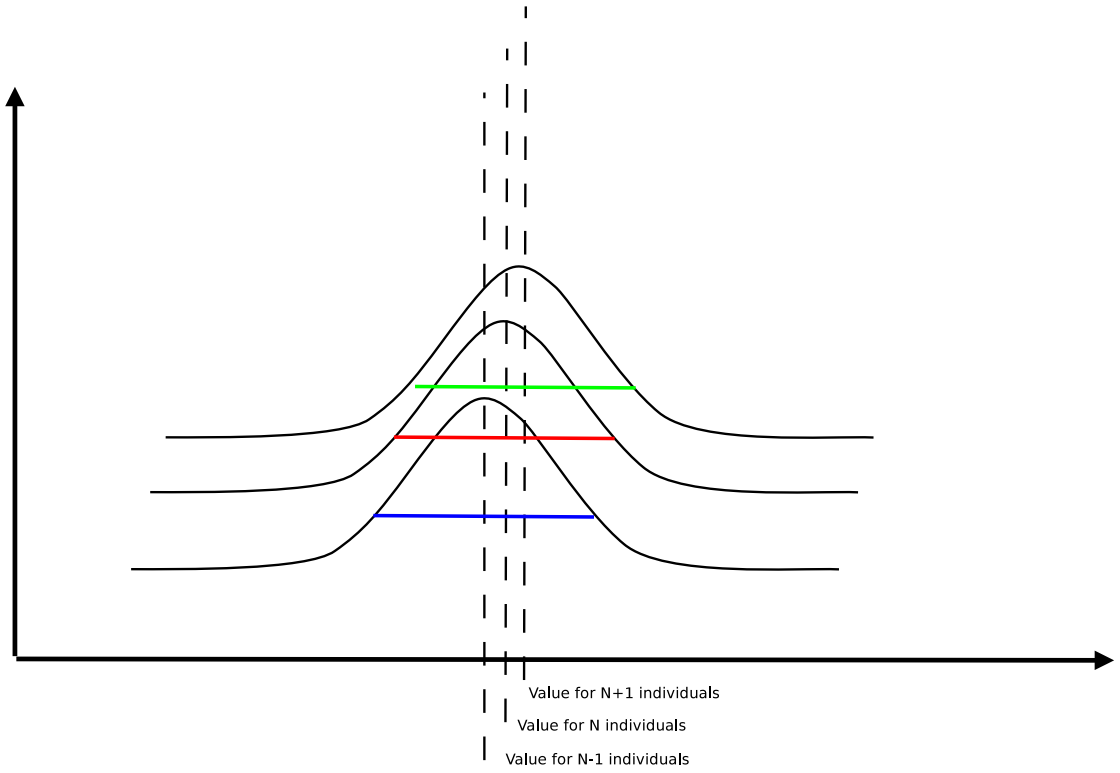


Figure 3.2: Visual representation of differential privacy. Three aggregation values are represented: one for N individuals and two for the neighbouring sets (i.e. removing one individual or inserting one). The noise which is added creates a likely segment of values. As can be seen, these segments overlap for the neighbouring sets: this makes it harder to guess from which set the result originated. This illustrates the idea of privacy. We also loose track of the real value: this is the loss of accuracy.

more than by a ratio of $\exp \epsilon$. We then need to balance the levels of noise in order to define a compromise between privacy and utility. This balance is the critical point of Differential Privacy (see Figure 3.2). An example of this can be found in the pharmacogenetics study published in [11]. In this publication, the authors try to establish what the correct dosing of warfarin would be, using differential privacy as a protective method for the privacy of the patient. Their conclusion is that if the trade-off between privacy and utility is set in such a way that we do prevent attacks, the dosing is so far off that "patients would be exposed to increased risk of stroke, bleeding events, and mortality". This is of course the worst possible outcome.

However, this is not the only challenge when constructing a differential privacy defensive mechanism. The first issue is that in order to correctly build the trade-off between utility and privacy, one has to make assumptions about the data. This was introduced by Kifer and Machanavajjhala in [12], in a paper with a much broader scope than GWAS.

The main issue in GWAS studies is that there are so many different variables that the possible metrics far outreach the number of patients. This huge number of possible outputs leads to the need of increasing the levels of noise added to the results. Researchers attempted to reduce the number of such outputs. In this line of work, Bhaskar et al. ([13]) propose methods to discover the K most frequent patterns in a genome association study (here the noise is in the form of the changes introduced in the returned patterns). But, as Johnson et al. point out in [14], the difficulty resides in that the correct number k is hard to know beforehand.

Although this line of study has these intrinsic flaws, the research on it has been strong in the field of protecting aggregation of genomic data. The research interest concerns, of course, the metrics which are most relevant for a genomic association wide study, for example ways to reveal minor allele frequencies, or the chi-square statistics as in [15]. Such methods were further refined: for example the work in [15] was extended in [16] with the introduction of χ^2 statistics with variable numbers of individuals in the case and control groups.

One solution to ensure high privacy and high utility at a time is to increase the number of participants in the study. However, this is costly, and we prefer to find other approaches. In [17] we are introduced to the idea that by better assessing the attackers' knowledge we will be able to use less noise: we have the required privacy levels, without the loss of accuracy. Whereas we assume in other differential privacy studies that the attacker has complete knowledge about both the data used in the aggregation and of his victim, Tramer et al. introduce the idea that just for some individuals the attacker will be entirely sure that s/he is or is not in the data set. For the others the attacker bounds probability of the individual being in the data set in the range $[a, b]$.

As we have said, the most obvious drawback of differential privacy is that we have done costly extraction of data, and we then add noise loosing accuracy. Dwork's answer in the Standford's GAPP conference ([2]) and in [18] is that we can use differential privacy over a hold-out set, when training some kind of algorithm. A hypothesis is built using data which is entirely accessible, but when the researcher wants to test it, he goes to the hold-out set where differential privacy will prevent any over-fitting. In this situation, the noise is even turned to an advantage. Such an approach adapts to the will of the individuals: in the case where they accept to participate in the study they can prefer to hide in the multitude or give a much more extensive read permission.

3.3 James Watson

In 2008, a team of researchers sequenced and published the DNA of Dr. James Watson. He was concerned about publishing portions regarding hereditary risk of Alzheimer's disease: he asked that his APOE gene (linked to Late Onset Alzheimer Disease or LOAD) be removed from the publication.

This privacy measure was defeated by Nyholt et al. in [19], or at least it was indicated how to defeat, since the authors did not want to go against the will of Dr. Watson. They show how using linkage disequilibrium in the surrounding area, one can accurately infer the data which was hidden. This word of caution led to erasing 2-Mb worth of information around the region of interest.

In 2015, Samani et al. ([20]) proved that such an attack could also be carried out using another mathematical approach. The authors build a Markov model which tries to represent the probability to see a given allele, on the basis of the knowledge about the previously seen alleles.

3.4 Homomorphic encryption

We are considering how to execute some algorithm on information which has to be private due to its sensitivity. This use case is not a specificity of the genomic data. Quite on the contrary, this feature is needed in many different fields: for example a web search engine where the server could not know what the submitted query is, or just a database with very limited access to the query being run and the data which is stored (e.g. CryptDB).

The idea behind homomorphic encryption is quite straightforward: we want to send an encrypted input to a machine which will execute a task on this input and reply with a response which is unintelligible except if one has the key. This is represented in a visual way in Figure 3.3.

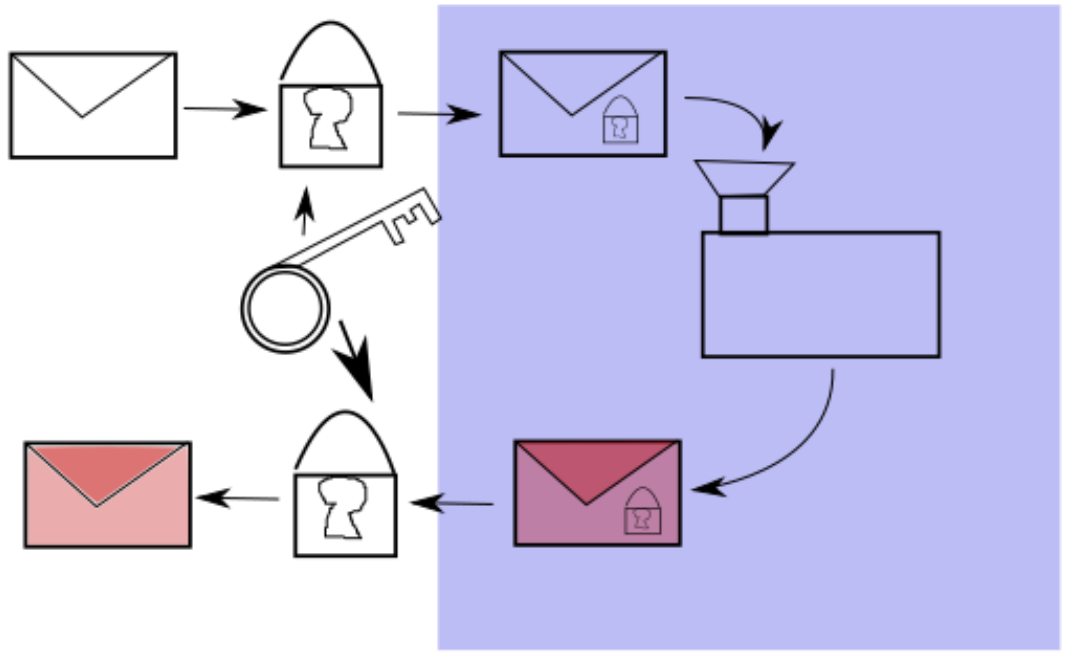


Figure 3.3: Remote computation using homomorphic encryption. White envelopes are the input of the program, red envelopes are the output, while the blue region is a possibly non-secure region such as the cloud. We encrypt the input and send it to the service, we obtain the encrypted output, which we need then to decrypt.

In order to achieve this we need the following mathematical property for the encryption and decryption functions (E and D respectively) and the function f being executed, x being the input and y the output:

$$y = f(x) \tag{3.1}$$
$$y = D(f(E(x))) \tag{3.2}$$

In other words, it should be the same to execute the algorithm on top of the original input as to execute it on encrypted content which is then decrypted.

With the ElGamal encryption scheme we can give an example quite easy to follow. We name the generator g, the secret key x, a random number r and we define also $h = g^x$. The ElGamal encryption of a message m is then

$$E(g^r, m \cdot h^r) \tag{3.3}$$

Therefore, if we have two separate messages and we encrypt both, we obtain two cyphers (C_1 and C_2) equal to:

$$C_1 = E(g_1^r, m_1 \cdot h_1^r) \tag{3.4}$$
$$C_2 = E(g_2^r, m_2 \cdot h_2^r) \tag{3.5}$$
$$\tag{3.6}$$

If we now multiply both we obtain the following result:

$$C_1 \cdot C_2 = E(g_1^r, m_1 \cdot h_1^r) \cdot E(g_2^r, m_2 \cdot h_2^r) \tag{3.7}$$
$$= E\left(g_1^r \cdot g_2^r, m_1 \cdot h_1^r \cdot m_2 \cdot h_2^r\right) \tag{3.8}$$
$$= E\left(g^{r_1+r_2}, (m_1 \cdot m_2)\, h^{r_1+r_2}\right) \tag{3.9}$$

From the last line we can clearly see that the result of multiplying two cyphers is a new cypher encrypting the multiplication of both messages (using the sum of the random numbers of the original messages).

The idea is that a cypher which would be able to accept any given number of homomorphic sum and multiply operations would allow for the computation of any kind of circuit, i.e. program. Until lately this was clearly out of reach, but recent breakthroughs have brought us closer to it ([21]). However, executing a multiplication on the homomorphically encrypted data is still costly in time.

In order to overcome these technical difficulties, authors have, for example, built strategies which could to some extent be assimilated to a variation of homomorphic encryption. The authors of [22] have clearly taken this option. In their article they describe how to make usage of the cloud computational resources to compute the alignment of the reads with the reference genome. They rely in fact on two distinct

clouds/groups of computation: a public cloud and a private cloud (where data can be executed without worries of being hacked by the service provider). In order to use the cloud without letting the reads be readable in the clear, they compare them with the references using hashes, searching for the most similar regions. This process is not based on the entire read, but rather on a smaller portion of it. The idea is that the public cloud retrieves possible location candidates, and then the private cloud checks whether those regions really match properly: this approach is based on the "seed-and-extend" method. Based on views expressed during the first GenCom workshop, we could ask ourselves whether the overhead due to communication could nullify any advantage. According to the authors, however, the communication overhead is "rather small". Using a 40MBps link, and due to the fact that the reads are not sent entirely but rather hashes of portions the time needed for the actual transfer was shorter than what we could have anticipated.

In order to perform alignments, the usual approach is using edit distances. The objective is to compare two sequences and determine how close they are from one another (an insertion, a deletion or a mutation are all one difference). A method able to compute this securely on the public cloud would be another path for the use of commodity computation clouds for the alignment of DNA sequences. The authors of [23] describe such a method. They avoid to use a full-homomorphic encryption in order to escape the hurdles which come with that. However, even when eluding the need of bootstrapping (technique used to correct possible mistakes in the homomorphically encrypted data after a multiplication), their algorithm is very memory intensive. On top of that, it is also very slow: they estimate that if the memory problem was resolved, they could compute the comparison of two 50 nucleotides long DNA strings in one day (while the length of the genome is in the order of magnitude of the billion). It is important to note that both strings are encrypted with the same key since the contrary is never stated. One is left wondering whether, in the case of using this proposal in a real production environment, we would have to resend the reference genome to the cloud encrypting it with another key. In the previously described paper ([22]) this was avoided by having the same hash values for everyone. The solution might be to use the same key for all the patients of a same institution.

Yasuda et al. ([24]) also focus on recognizing regions of the DNA, but they concentrate on pattern matching. Their method returns the Hamming distance, executing the computation securely on the cloud using Somewhat Homomorphic Encryption. We should note that in the system they describe, they also assume that the DNA and the pattern to search are encrypted under the same key. They consider the doctor or institution starting the computation as being a trusted party. Based on previous publications on the matter, they also consider how to pack the data in less space in order to enhance communication speeds and performance

on the cloud. However, what they propose is very much tailored for the task at hand (i.e. Hamming distance), and for the needs of homomorphic encryption: they are not attempting to use repetitions in the data to spare some size, but rather combine different information in each encrypted polynomial involved in the computation. Therefore, it seems quite impossible that their proposal could be relevant as a practical storing strategy for genomic information.

A quite similar problem to resolve is the question whether there is a specific marker in a given DNA string. Both ideas are related: both are ultimately the comparison of two strings. Cristofaro et al. ([25]) propose a solution to this new issue. They present a secure test for the case where we know that the presence of a given marker at a given location is a factor which increases the likelihood of a disease. What they propose is twofold in fact: on the one hand nobody other than the person in possession of the DNA should learn which markers were detected, and on the other hand nobody other than the provider should know the test being performed (i.e. which substring is searched for, and at which position). The motivation behind protecting the test is plainly defending the interest of the Intellectual Property (IP) of the company whose test is being executed.

Barman et al. in [26] also contemplate the need to possibly protect the IP of a company. What they consider is the case where there are effectively two parties: first a Data Center attempts to defend the patient's data it holds, and second a Medical Center executes a request to this data in order to obtain a measure for a given patient but wants the request to remain secret. In the construction Barman et al. propose, the Medical Center sends an array of weights: the response of the Data Center is the scalar product of the SNPs and the weights. In this model, both players can be the attacker or the defender. The attack of the Data Center is merely looking which are the SNPs which are requested. In order to defend itself against this, the Medical Center adds dummy weights. It then generates, for each weight, what the authors call a commitment: this can be viewed as a hash of the weight. The Data Center then selects randomly two commitments and asks for the weights: the Data Center checks the correctness and whether the numbers might indicate an attack of the Medical Center. Such an attack could be either using many zero weights, meaning that the end result is equal to the value of the only non zero-weighted SNP, or using a sequence of powers of the same number. If the Data Center has suspicions, this step can be repeated. Ultimately, the scalar multiplication is executed homomorphically. As the authors state, the Data Center could deviate from the protocol and request more verification rounds, but the Medical Center can detect this and abort the procedure. Similarly, if too many weights appear to be non-legitimate, the Data Center can decide to stop the query.

Up to now we have only described articles considering searches on an encrypted

DNA sequence, or scalar products. These are not the only uses of homomorphic encryption. For example in [27], the authors describe a method of computing the χ^2 statistic securely on the cloud. As seen before, they also argue that they reduce the size of the encrypted data which is needed, but this procedure is once again intended for the use in homomorphic encryption and involves the way in which the whole data is packed in a polynomial. Interestingly enough, this paper proposes a solution for the case of a secure "meet in the cloud": neither bring the computation to the data, nor bring the data to the computation. One could argue that the previously mentioned papers do not follow neither of these two concepts since it is one party which sends both the encrypted data and computation to the cloud, but Lu et al. take yet another approach. In this case, they use a secret and a public key: every participant in the study (a repository with genomic data either from the case or control group) encrypts his data with the public key and sends it to the cloud. The cloud then computes the result which is subsequently decrypted by the research team using the secret key. As could be expected due to the fact that just one party, the researchers, decides the key or key-pair which will be used, there is a risk that they could obtain the data sent by the repositories: if the cloud and the researchers collude, the cloud could send the encrypted inputs to the researchers who could easily decrypt the computation using the secret key.

The χ^2 statistic is commonly referenced as a go-to measure in Genome Wide Association Studies, but there are other methods such as the Logistic Regression method. This approach is maybe even more useful since it can accept other factors to explain a disease such as variables indicating the exposure to certain risk elements. The authors of [28] describe a way to compute such an experiment on the cloud, protecting the data once again with homomorphic encryption. In the model they present, there are two parties: one party with the information about the SNPs of a set of patients, the other party with information about the disease status and possible grouping of each patient. The authors give gender and ancestry as examples of grouping. As in the secure computation of the χ^2 statistic in [27], there is also the use of a key pair.

The result of such a Logistic Regression can also be used in a homomorphic encrypted computation. A team at Microsoft released a library for the development of homomorphically protected computations. The use of this library called SEAL (Simple Encrypted Arithmetic Library) is described in [29], and an example which is provided within this manual is the computation of a decision based on Logistic Regression. This use case is quite interesting for the protection of the genomic data of one individual. Until now, we have seen examples where the whole collection was secured, in other words the whole data for a Genome Wide Association Study was secured, but there was still one party who had access to all the information. Executing an algorithm for just one patient is not at all the same approach as for

a GWAS, but SEAL shows that there is at least one use case where homomorphic encryption does defend the interest of one person.

Chapter 4

LITERATURE REVIEW THROUGH THE LENS OF MPEG REQUIREMENTS

In the process of constructing a secure file format for genomic data, the MPEG workgroup has defined a list of requirements that such a format should consider [30]. These requirements concern different aspects in the use and life-cycle of the file format and are listed in Table 4.1. Additionally, other requirements were proposed in [31] and are listed in Table 4.3. Alongside the ISO/IEC MPEG initiative, the Global Alliance for Genomics and Health (GA4GH) also works on defining strategies to allow compatibilities across different beacons and to define standard ways to query these beacons [32]. These goals have to be met respecting security guarantees, which leads to the requirements defined by the Global Alliance for Genomics and Health, which are listed in Table 4.2. Tables 4.1, 4.3 and 4.2 group together the requirements, a short explanation, and a list of publications which contribute to the point.

As we have seen, current research places its focus on the construction of secure architectures and protocols for a privacy-aware usage of genomic data. This leads to the fact that some points considered by MPEG and GA4GH, such as the need of integrity, are not covered by present research. However, some of the points addressed are an area of active work.

For example, both groups specify as a requirement that the individual has to be made aware of the usage which will be made of the data. This point, which is also strongly related to the need of authenticating the parties and controlling their behaviour, is treated in many publications and presentations as indicated in the tables. Shelton for example presents at the GAPP Conference 2016 at Stanford [2] a model according to which research groups have to explain their intentions to the individual concerned in order to obtain the user grant. This principle is also

fundamental to the concept of Dynamic Consent (i.e. the on-demand informed consent, in opposition to the broader and too extensive grant currently asked for). We also give a pointer to De Cristofaro [33] who complements this aspect by his study on the individual's fear about the usage of his/her data.

Another point, which is less covered by the publications, is the protection of genomic data "on the hard disk". Obviously the need for security is a central point of study, but except for Hubaux [2] no study seems to consider from which file the data is read. The main publication trends concern rather the result obtained from the data, as we saw in section 3.2. In fact, many publications deal with differential privacy, which does not correspond to a requirement formulated by either MPEG or GA4GP.

On a similar note, the security requirements concerning the transport mechanism do not have a counterpart in current research, except for the studies on homomorphic encryption. Although homomorphic encryption does not transmit the file in such a way that no other party can make use of it, it nevertheless ensures the security of the data conveyed (see section 3.4). Besides, homomorphic encryption is also relevant to the requirements concerning the use which is made of the data: since the data is unreadable, no action other than the requested one can be performed. Other relevant ideas for these requirements are the policy engines such as those described by Dave Maher, Carl Gunter and Robert Shelton in [2].

In the requirements proposed to MPEG [31], we find more matches with the current research directions. For example, the consideration of possible damages for blood-relatives in the case of a leakage is dealt with in the article by Humbert et al. ([34]) on how to include the will of the relatives when deciding what mutations should be published. Their study takes into account the preference of the individuals and of their blood relatives, the genetic information they have published, the possible leakage regarding diseases and the utility of the information for research in order to build an optimization problem with the constraints derived from the previously enumerated criteria to decide which SNPs are the most interesting to publish.

At this point we should state that against what the intuition could be, it seems that reidentification through SNPs is not a common thing: the techniques used for the identification of an individual through his/her DNA footprints are based on other properties of the DNA, namely Short Tandem Repeats (short strings which repeat frequently), which are mainly located in non-coding regions of the overall genome. This information has multiple implications for the task at hand. On the one hand, there might for the moment be a lack of real understanding of how many SNPs are needed to identify a person (although, as we saw with [9], some consider this use case). On the other hand, when addressing the issues of leakage, access rights and identification, we cannot consider the non-coding regions

as meaningless, quite on the contrary.

Finally, Goodrich's publication ([35]) on an attack based on the Mastermind board game is relevant for the different requirements concerning authentication and information about the use of the data. In the situation described by Goodrich, there are two parties: one party with a DNA record which should remain anonymous, and the other party with a pool of DNA records with their identities attached to it. The first party wants to find the similarities between its DNA records and each record in the pool. The author describes how to build on the database side an attempt to recover the data, assuming that the only information being revealed is the similarity score. This attack proves that the currently formulated requirements might not be sufficient: repeating a task which is in principle permitted could eventually lead to a successful attack.

Requirement	Rationale	Publications
The compression process shall support the assessment of integrity	Integrity check shall be possible by providing appropriate information.	
The solution shall allow conveying information enabling data protection	Ability to prevent unauthorized access shall be available. Information needed for protection of data (control for access, modification, publication, etc.) shall be conveyed.	Presentations at Stanford workshop (Hubaux, Maher, Gunter, Shelton, Lauter [2]); homomorphic encryption ([22, 23, 24, 25, 26, 27, 28, 29])
The solution shall allow conveying information enabling accountability and traceability	Data access and manipulation shall be traceable together with the identity of parties having access to data. Information on how to verify integrity and authenticity of the data shall be conveyed.	Presentations at Stanford workshop (Malin, Maher, Gunter, Shelton [2]) [35]
The solution shall allow conveying information enabling transparency	How and for which purpose the information is used shall be known. Usage restriction shall be applicable to the data.	Dynamic consent ([36, 37, 38, 39, 40]); presentations at Stanford workshop (Malin, Maher, Gunter, Shelton [2]); [33]

Table 4.1: List of MPEG's requirements [30]

Requirement	Rationale	Publications
Identity management	Identity of individuals and software accessing the data have to be authenticated, but also the files themselves.	Presentations at Stanford workshop (Hubaux, Maher, Gunter, Shelton, Lauter [2])
Authorization management	Mechanisms have to be built to determine whether the access request can be granted.	Presentations at Stanford workshop (Hubaux, Malin, Maher, Gunter, Shelton [2])
Data Security Safeguards	Only authenticated and authorized users are granted access and they must ensure that authorization rules attached to the data are respected in all their uses. The service provider also has to maintain all logs needed for an audition. Finally, the integrity and non-repudiation of the data has to be guaranteed.	Presentations at Stanford workshop (Hubaux, Malin, Maher, Gunter, Shelton [2]), dynamic consent ([36, 37, 38, 39, 40])
Cryptography	The data should be protected through strong encryption, compliant with relevant requirements.	Shelton's Stanford presentation [2], homomorphic encryption([22, 23, 24, 25, 26, 27, 28, 29]), [35]
Physical and Environmental Security	The storage and processing, either on location or provided by a third party, should be protected according to applicable laws.	Presentations at Stanford workshop (Malin, Shelton [2])
Operations Security	Security and privacy practices should be made public and available to all parties concerned.	Dynamic consent ([36, 37, 38, 39, 40]); presentations at Stanford workshop (Malin, Maher, Gunter, Shelton [2]); [33]
Communications Security	Each transmission of genomic and medical data should be protected with secure communication technologies.	Presentations at Standford workshop (Hubaux, Maher [2]), homomorphic encryption ([22, 23, 24, 25, 26, 27, 28, 29])

Table 4.2: List of Global Alliance for Genomics and Health's requirements[32]

Requirement	Rationale	Publications
Integrity	Ensuring the integrity of the file	
Phenotype inference	It should not be possible to infer phenotype traits from the DNA records.	[9], [3], [34]
Risk for blood relatives	A DNA records leakage could have consequences with possible consequences for the victims' offspring.	[34]
Anonymize the data	Although the DNA has intrinsic identification properties, the data should be as anonymized as possible.	Presentations at Stanford workshop (Maher, Gunter, Shelton, Lauter [2])
Need for consent	The consent should be built into the file.	Presentations at Stanford workshop (Maher, Gunter, Shelton, Lauter [2])
Data protection	Protect the stored data using techniques such as encryption or genome splitting.	homomorphic encryption ([22, 23, 24, 25, 26, 27, 28, 29])
Well-defined queries	Define mechanism to accept well-defined queries to the data stored in the file.	Homomorphic encryption ([22, 23, 24, 25, 26, 27, 28, 29]), presentations at Stanford workshop (Maher, Gunter, Shelton, Lauter [2])
Information withdrawal	The databases have to allow the withdrawal of information at any time.	Dynamic consent ([36, 37, 38, 39, 40]), presentations at Stanford workshop (Maher, Gunter, Shelton [2])
Participant-centred	The file format should be compatible to the use in participant-centred studies.	Dynamic consent ([36, 37, 38, 39, 40]), presentations at Stanford workshop (Maher, Gunter, Shelton [2])

Table 4.3: List of requirements proposed to MPEG [31]

Chapter 5

ONE POSSIBLE SOLUTION

As we have seen, genomic data can be used in different applications. The problem is that the non-legitimate uses of this information can seriously harm the biological owner of the information. By saying biological owner we try to establish that sometimes the legal owner of the data might be the research group or similar which sequenced the data. Despite the fact that they also have claims to make over the data, it is just correct to restrict their access to what they really need for the research and what the patient is willing to provide as data.

We are trying to define a format which allows to share genetic information, and we have just concluded that we might have the need to hide certain portions of it to the unauthorized reader. This responds directly to the second point stated in the requirements proposed to MPEG [31] (protection against phenotype inference), the third point (taking into account the risk of leakage), and finally the fourth (building the need for consent). We could either simply erase some portions of the file, or we could encrypt those portions. In the second case, some chapters of the file would be intelligible, while others would look like jibber-jabber to the reader. Both have advantages depending on the situation.

Reading current research we have seen that the cloud is expected to be a key platform in the use of genomic data. A file developed to hold DNA records should perform equally well in a cloud usage as in a local use case. However, if we think about the life length that such a file could have (being useful at the very least as long as the person is alive) and for how long nowadays individuals are able to store a file until it is lost due to lack of backups and similar, we are tempted to conceive the usage of such a file, from the point of view of the user, as being cloud-based.

On the basis of this idea and seeing what Private Access (Robert Shelton's presentation at Stanford [2]) proposes, we have to expect strong interaction through handheld devices when granting read requests for certain sections. Therefore we will strive to simplify as much as possible the information to be stored and transmitted in order to grant access to new portions.

In the case of a Direct To Consumer service where for some reason sending the data homomorphically encrypted is not feasible (for example the circuit which needs to be executed is too deep), we could generate a stripped-down file with just those portions which are needed. In such cases, it does not make sense to send the encrypted portions. In all formats we have seen (i.e. SAM/BAM and CRAM), this is no major problem : we have only to erase certain records. In the case of SAM/BAM it means erasing the lines of the readings we do not want to send, in CRAM we erase the blocks, or at lest the slices, which should remain secret.

Encrypting portions, however, does make sense in the case of an on-line repository such as a beacon. If it might be faster to send the key to decrypt a specific portion rather than sending the whole chapter, we would have spared some effort by including the unintelligible portions. The negative point is that it means that we need to be able to contact the sequenced person when a new request arrives.

The user story behind our proposition is as follows. The DNA of a person is sequenced by a medical laboratory. The sequenced person obtains his or her file and decides which portions should be readable and which not. Based on his/her decision s/he encrypts the secret portions and then sends the modified file to an online repository, beacon or other cloud infrastructure using DNA information. The services provided on the chosen platform might need to be given access to more portions of the file. In that case, the beacon will forward the grant request to the individual who might accept and reply with the information needed to have access to the portions in question. This workflow is summarized in Figure 5.1.

Subdividing the genome in meaningful semantic blocks is a practical path which has already been taken (e.g. the service provided by Private Access). The advantages are obvious: the patient can grant access to some of these blocks, and for others he denies the access. The genome provides us already with such distinct blocks in the form of genes. Now, two distinct challenges arise.

The first challenge is whether it should be possible to grant access to just a set of persons. From the most privacy-cautious point view, the answer is that such a rule would be the same as granting access to everybody. Once the read access has been given to one individual, there is always a way for this person to share the respective information without the grantor's consent or even knowledge. Even if borrowing DRM-technologies to multimedia, there is no guarantee that the grantee will not simply copy by hand and then recreate a new file with this information.

The second challenge is how to preserve the confidentiality of the non-disclosed blocks: as we have seen, statistical inference permits to deduce hidden alleles on the basis of the ones we have access to. We can employ two strategies for this: the usual opt-in and opt-out.

- Opt-in: we consider 'private' as the default state. In that case we will grant access on a per-block basis and inform the sequenced person deciding the

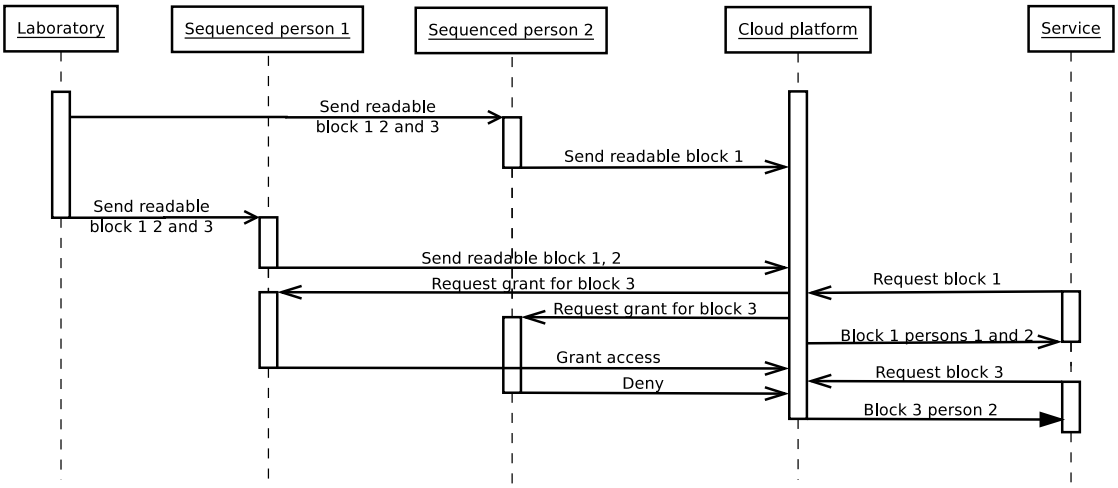


Figure 5.1: Workflow for publishing privacy-enhanced DNA. The DNA is published on a repository, taking into account the will of the patient regarding which regions are readable and which are not. If a group asks for information about regions which are encrypted, the repository will forward the request to the patient who can decide to give the key for that portion, thus enabling the read.

rules when, based on the knowledge we have, we assume that the privacy of some block(s) might be endangered. Then it is his/her decision whether utility or privacy should prevail.

- Opt-out: we consider 'readable' as the default state. The user will then add privacy rules denying access to some specific block. However, as we saw with the case of James Watson, the effects of these rules need to encompass more semantic units. Once again the user who generates the rules should be made aware of these conditions.

Using the idea of opt-in we might, however, find a middle ground which allows sharing genetic information with a subset of entities. We could construct multiple opt-in files which are then distributed according to the requests. In order to achieve this, we need to keep track of the different versions of the files which exist. The idea is to ensure that the crossings, i.e. portions which are in clear in two or more files, never allow parties to collude and recover enough information for a re-identification. In order to decide what enables an identification, we could use a kind of measure of identification potential. How this measure would be defined is still unclear, but as research in forensics starts using more measures based on SNPs and similar, this issue will be better understood. The overall idea of how to split the DNA into various coexisting files is summarized graphically in Figure 5.2.

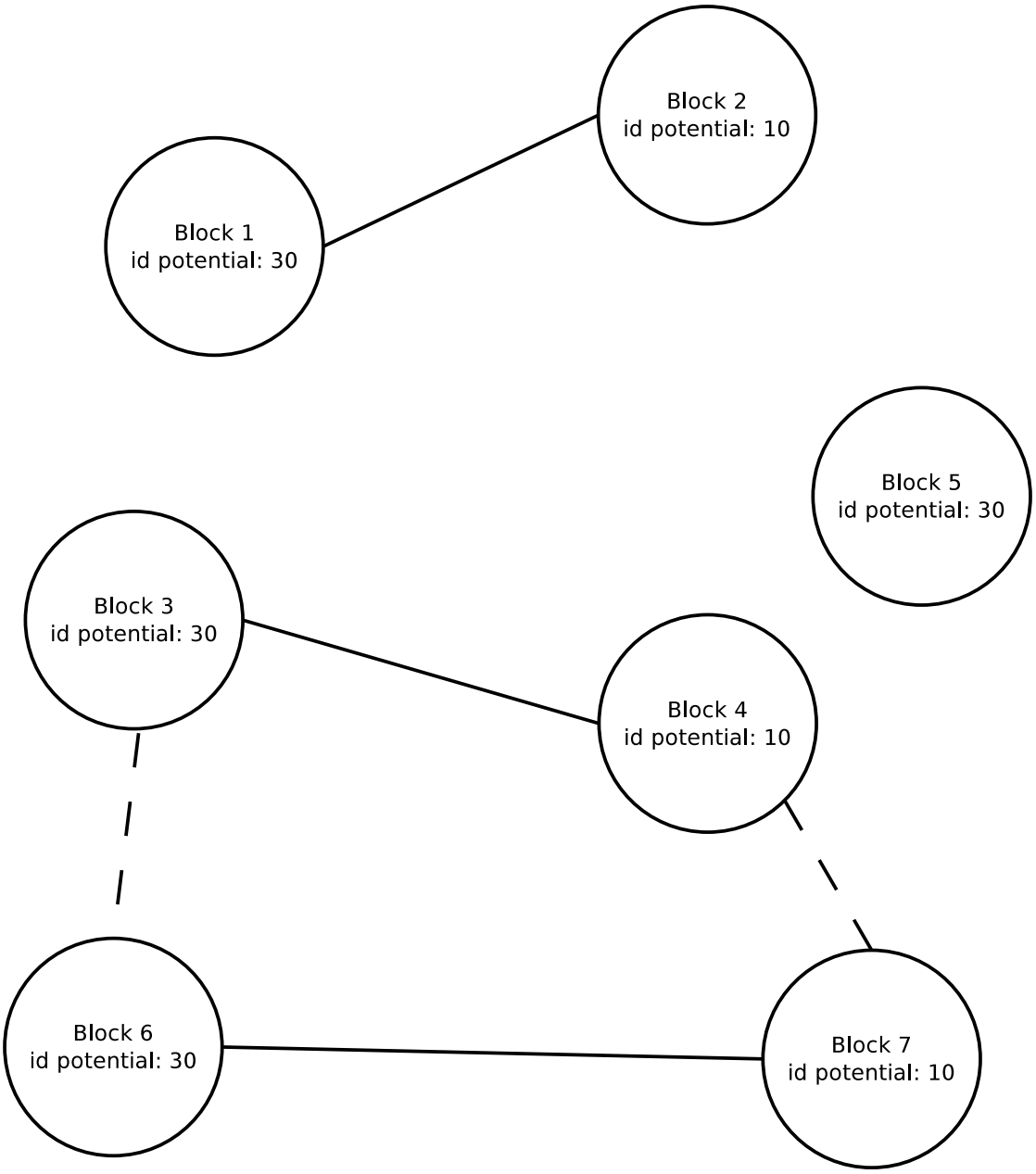


Figure 5.2: Theoretical example of DNA record split into 7 blocks, over 4 possible files. We grant each block an identification potential (id potential). In this example there are 4 files: each group of nodes connected through solid lines represents a file: blocks 1 and 2; blocks 3 and 4; block 5; blocks 6 and 7. For this example we fix the threshold for identification potential arbitrarily at 50, therefore the sum of id. potential of all files must be below this number. A case of special interest would be the question whether linkage disequilibrium (represented by the dashed line) allows to combine the id. potential of two files.

In any case, all of these methods require state-of-the-art knowledge about allele correlations and the gene functions. This might seem a challenge, but the attacker will be facing the same situation. Therefore we can assume that both parties will have access to the same information, allowing the defendant to correctly assert the risk that will be encountered.

Another challenge is how to define meaningful blocks of data. As we have said, genes seem a logical choice. In table 2.1, we have listed the different codons and what decisions they trigger. Some of them encrypt a STOP action, in other words we expect them to have the end of a protein recipe. The beginning of the coding region of a protein is, however, less clear: the role of the codon coding methionine is key (many proteins start with methionine), but other factors play a role in order for the cell to know where to start the synthesis. Furthermore, our understanding of the DNA might well evolve, leading to discoveries of new meaningful subdivisions in the DNA. Therefore, and although genes appear to be a sound choice, we need some other approach, since we are not ready yet to use a block subdivision entirely based on intrinsic properties.

On the other hand, we already know the positions of many different genes, and it would certainly be a waste not to use this information. Therefore the best solution might simply be to let both worlds coexist: if we develop a file structure allowing different lengths, then it is no problem to let both strategies exist side-by-side. For the uncharted regions, we will use either a default length or a default length improved with some heuristic (for example ending the block after a STOP codon), and for those regions where we have already an understanding, we use the current state-of-the-art. These different options are summarized in Figure 5.3.

5.1 Encrypting certain portions

We have a strategy to split the DNA into different blocks with which we can start implementing the selective protection. Let us begin with an easy file format based on either the FastA or FastQ format. In order to simplify it conceptually, we will first forget about the positioning information of the reads: we might just think that the reads are padded at the beginning and the end, letting us position them easily in front of the region they are aligned to. Furthermore, let us consider that the reads have the same length as the block they belong to (this means that we do not consider insertions or deletions but rather only modifications). We define now a key k_i for the block i, and an encryption function $E(d, k, s)$, where d stands for the data being encrypted, k for the previously defined key, and s for read identifier. The length of the output of E should be equal to the length of d. The counterpart of $E(d, k, s)$ is the decryption function $D(c, k, s)$ where c stands for the cyphertext. At this point, we expect the function E to be able to adapt to

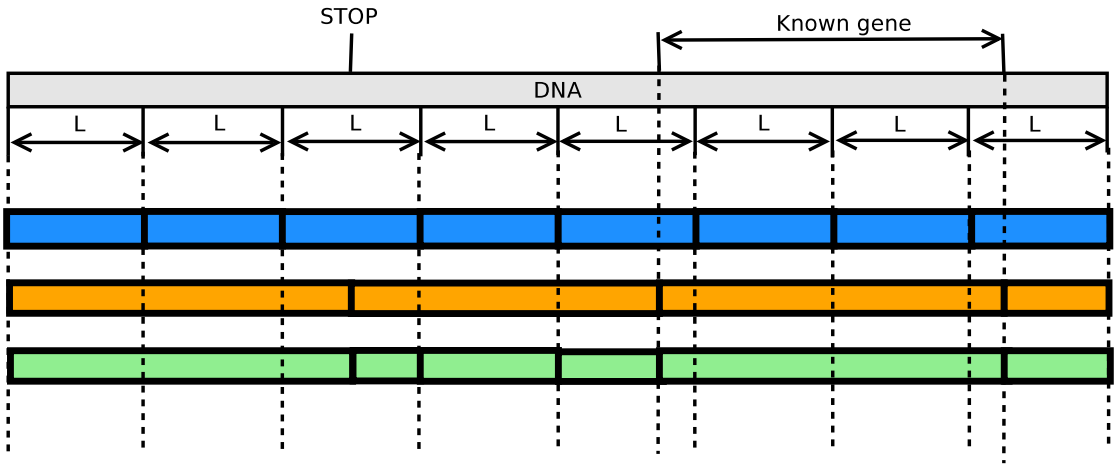


Figure 5.3: Possible block division of DNA: in this theoretical portion of DNA we have one STOP codon, one known gene, and uncharted material. The three rows depict possible subdivisions. The first row uses fixed length blocks. The second tries to use some heuristic: e.g. blocks span until STOP codons, known genes are used as blocks, and the unallocated information is put in extra blocks. The third row is based on the second, but limits the size of blocks of uncharted nucleotides by using a maximum fixed size.

content of any length. This would resolve the problem of having blocks of different lengths. In figure 5.6 we summarize these different points in a visual manner.

Our objective is to encrypt just certain blocks: this step is now easily achieved by encrypting each of the reads of a secured block with the corresponding key k_i and its read identifier (i.e. the read id s). Using some solution to encrypt differently each subdivision is a commonly known necessity in order to avoid the so-called Electronic Code Book where one can readily see which blocks are the same without even having to decrypt the file. We avoid this problem by using the read id s as source for the differences. If we are now asked to grant the read permission for a given block, we just need to send the key k_i and possibly s.

In other words, if we want to encrypt a block i, we replace each of its reads with the cypher obtained after encrypting them with function E.

5.2 Decrypting a portion partially

Let us introduce a new requirement for our security mechanism. We now wish to be able to decrypt some portion of the block. In other words, we would not receive a grant request for a whole unit of DNA, but for the region in between reference nucleotides i and j. Introducing this possibility could have important

benefits: more fine-grained read requests would be less frightening for individuals, and it would allow us to correct division mistakes made while creating the file. Let us imagine for example that after having generated the document a new gene is discovered which is within one block. By granting access to the whole block we grant access to more information than needed, whereas this new requirement helps us keep one additional fraction private.

Our encryption function needs, however, to be adapted to this new use. The Advanced Encryption Standard (AES), currently an extremely common encryption scheme, uses permutation steps and similar which would not allow us to simply decrypt one part of the cypher. What we need in order to achieve this goal is the use of something more similar to the one-time pad. This encryption strategy works on a bit level, the key is as long as the plaintext and consists of random bit values. In order to encrypt and decrypt we xor together the plaintext and the key. If we give a portion of the one-time pad and the location for which it is intended, the encryption/decryption still works. In other words, the encryption of each bit is independent of that of the others.

The main drawback of the one-time pad scheme is the need to provide a random key as long as the actual content. In order to amend this, we can use functions which generate random-like output such as the AES scheme. The tweak resides in the fact that we do not use the plaintext as input of some sort for the encryption function any more: only the key and some other parameter are involved in the generation of what is ultimately the one-time pad.

There are two methods of using a block cypher in such a one-time pad generation framework: the Output FeedBack (OFB) and the CounTeR mode (CTR) (which can be seen in Figures 5.4 and 5.5 respectively). Let us first think about what we have to expect for the reads. In the case of no mutation whatsoever, the read will be a copy of the reference (in the case it includes quality metrics, it is less of a copy, but it is still very predictable). If there are mutations, they will probably be due to ancestry, and in that case the content of the read is very much predictable. Even in case there are less common frequencies, the context or published statistics might also help to figure out what the plaintext might have been. In other words, we have to consider that this encryption is under a very severe risk of known plaintext attack. A plaintext attack is the attempt to recover the key and other parameters of an encryption function given a set of known pairs of cyphertext and plaintext encrypted using those parameters. The resilience of an encryption function against such a threat is a well-known criterion for the suitability of that function. When choosing the actual function E, we therefore need to take into account the current knowledge in order to reach an informed decision.

If access is granted to the whole block, the previously described methodology still works: we share the key and either the initialization vector or the counter

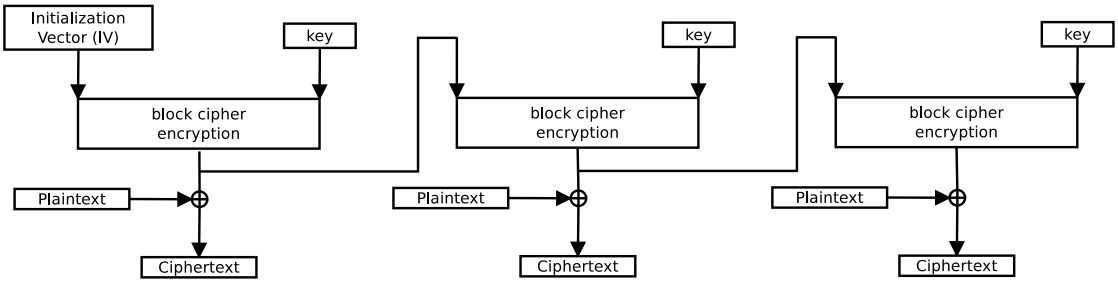


Figure 5.4: Output feedback construction

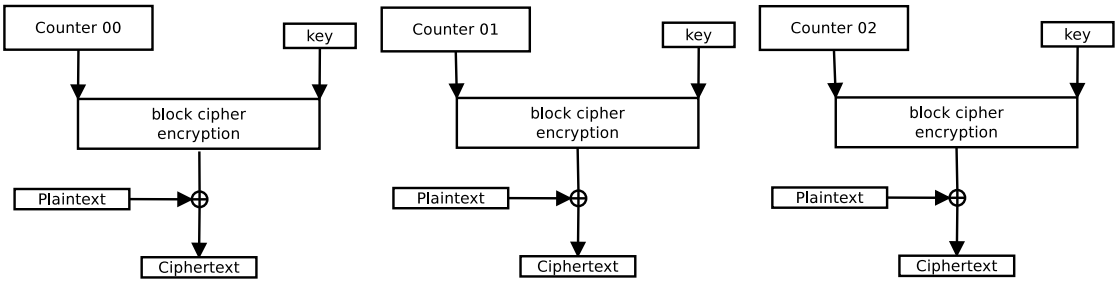


Figure 5.5: Counter construction

generation method as required. In the case of sharing a specific subset of a block, however, the grantor generates the required portions of the one-time pad and sends them to the grantee, who can then decrypt those portions. The drawback of such a method is that we possibly need to send far more bits, which means that in order to speed up distribution the block method should be preferred.

As we said before, someone attempting to break the security may take advantage of the fact that the content is predictable. If the attacker has now even access to the read grant, the information about the plaintext he does not dispose of is even less extensive, which puts even more stress on the required defence mechanism against a known plaintext attack.

5.3 Including insertions and deletions

Up to this point we have avoided the challenge of insertions and deletions. If we take them into account, the size of the read will not match the size of the reference block. Therefore, even without any access to the decrypted content of the file, one might guess the presence of certain mutations. With deletions we might avoid the problem by including a symbol for a blank space. However, such a straightforward solution does not exist for insertions.

In cryptography, when a plaintext is too short for an encryption scheme, it is

padded at the end with a content easy to distinguish. This increases the size of the cypher, but without this procedure the encryption function could not be applied.

In our case, it might be useful to build upon this idea and to include a pad of predefined extra slack to all reads. This extra space will help us make any insertion invisible without decrypting the entire read. However, there are some severe inconveniences attached to this procedure. It might be hard to predict the size of the required slack since it needs to contain all possible insertions for that block for every individual. It is not sufficient to use some value greater than the maximal number of insertions we see for that individual, because doing that would not protect against comparing with some other file. The second disadvantage is that the file becomes heavier unnecessarily.

Let us suppose that based on some heuristic we have decided what the appropriate slack could be. We then have to decide where it is more logical to write the information about the deletion. Let us imagine that we have the following scenario:

```
reference:
TTAGATAAAGGA_TACTG
read:
ATAGATAAAGGAATACTG
```

We might publish the insertion as extra information at the end (indicating the point of the insertion and the inserted nucleotide), or indicate all nucleotides in the read sequence:

```
                          12345678901234567
reference:                TTAGATAAAGGATACTG
insertions at the end:    ATAGATAAAGGATACTG#12A
read sequence:            ATAGATAAAGGAATACTG
```

With the first solution we do not introduce any indexing problems: we maintain the original nucleotide ids from the reference, but we have introduced extra weight. For the second solution, the problem resides in the fact that if someone queried for any index after the insertions, he would see a change in the nucleotides whereas in fact the reads are just shifted in respect to the reference.

Trying to resolve the indexing issue might in turn bring other complications. We have two clear options. The first one is to add a protected metadata which explains how to convert from the reference index to the read index. The motivation for protecting the metadata is that this information allows to see readily whether there is an insertion or not, even if the reads are encrypted. The inconvenience implied is that the form of the indexation is a very important aspect of making the file — or at least the sections which should be accessible to everyone — useful.

All in all, it might be easier to include the insertions at the end, and consider that the default access method should apply to the whole block.

However, this solution focuses on just one kind of mutations: those involving just one nucleotide at a time. In some cases mutations affect whole regions: whole sections are entirely deleted, inserted, duplicated or simply shifted. When a section is entirely deleted, it might be easier to rely on a symbol which indicates the absence of that location, as we have seen previously. In the case of such insertions, however, the indication at the end of the read is not adapted any more. On one hand, we would have to store all its nucleotides as insertions into the previous block, losing thus the bounded location on the reference genome. On the other hand, the proposed notation would be extremely heavy, since we would have to include the position for each inserted nucleotide.

5.4 Implementation of contact information

The possibility to contact the individual should be a key feature. In the solutions overviewed in Chapter 2.3, there is already the idea that the genomic records are linked to one profile/person. Robert Shelton claimed in his presentation at Stanford ([2]) that the contact information was just another portion which could be made public or not. This could also apply to the file format we might propose, but we could rather be interested in allowing the recipient of the information to contact the sequenced person without being able to know who s/he is, mainly because it is the ground stone for any future grant request.

A first approach could be to have a trusted party which generates random but unique identifiers for each file. This identifier would be integrated to the header sections as in either SAM/BAM or CRAM, as just one more field. When a read request needs to be sent, the research group would contact the trusted party and ask it to forward the request to the persons behind those identifiers. As mentioned before, one individual could generate files with different splits of his/her DNA record, therefore one contact information could be linked to more than one identifier.

However, this solution allows collusion among parties, making the reidentification and the obtention of far completer DNA files trivial. One way to avoid this might be using the trusted third party as a simple mail box: the person creates anonymously an address for grant requests. With such a solution the inference is less evident, but there might still be methods to infer that the same mail boxes are owned by the same individuals, for example through the analysis of Internet addresses.

This type of issue is faced in other projects, and we might simply use one of the existing solutions such as the blockchain. The blockchain was developed for

the digital currency Bitcoin, and acts as a distributed and public ledger. The synchronization among parties is achieved when one of the parties is able to find a solution to a computationally hard problem which validates the previously seen transactions. There are already use cases of variants of the blockchain: for example, for the Twister project [41], a distributed twitter equivalent aiming at maximal security, the blockchain is used to register the public key of new accounts. This could be an idea for guaranteeing the kind of anonymity we are striving for: we generate randomly an id, and publish a public key related to it; then the grant requests are published on a public and well-known whiteboard, and everybody who wishes to grant that read just needs to send the information required signing it with the secret key. Obviously such a message would also be encrypted in order to guarantee privacy.

However, this solution is not good enough. To begin with, the blockchain relies on rewards. As we said, there are computationally intensive tasks being executed. There would only be downsides in performing such tasks (need for dedicated computers and electricity cost) if it were not for the rewards: in the case of Bitcoin the reward consists in Bitcoin coins, in the case of Twister in promoted messages (equivalent of advertising messages in Twitter). It is very unclear what could be the reward in the case of a platform meant to guarantee anonymity. Building on this, the other challenge is whether the security requirements could be achieved: there is a risk if we cannot sum enough parties to participate in the computational task. In such a case, a subset could collude and take over the whole decision process if it adds up to more than half the computational power. If there are no incentives, it is just too likely that not enough parties would participate, which would lead to security problems.

The best solution might be to simply apply one of the easiest options. The patient generates a token at some kind of Certification Authority: the identification number is random and guaranteed to be unique, and a public key is published. Research groups publish on well-known locations their request for grants, and if the person is interested in providing it, s/he sends the requested information signing it. The research group only needs to verify the authenticity and can use the newly granted information.

5.5 Securing the SAM format

The read length will never match the length of the actual block. At the moment we assume they do, which means that we have to introduce padding increasing the weight of the file unnecessarily. Furthermore, we assume that our encryption method E will adapt to the size of the content, which is usually not the case (e.g. in SAM). In order to solve this problem we have to change the conceptual

representation of the file. Instead of thinking about each read as a length unit for encryption, we should rather concatenate all the reads, and encrypt that content. This allows us to accept reads of different sizes.

In the current state of sequencing methods, the reads cannot be focused on a given region, and the length of each read depends on the technology, not on the intention of the technician performing the study. This could help us motivate that there is no need in protecting the 'metadata' of this read. Based on the previous example of a SAM file, we now colour the regions of interest which should be encrypted according to this reasoning.

```
@HD VN:1.5 SO:coordinate
@SQ SN:ref LN:45
r001  163 ref  7 30 8M2I4M1D3M = 37  39 TTAGATAAAGGATACTG *
r002    0 ref  9 30 3S6M1P1I4M *  0   0 AAAAGATAAAGGATA   *
r003    0 ref  9 30 5S6M       *  0   0 GCCTAAGCTAA       * SA:Z:ref,29,-,6H5M,17,0;
r004    0 ref 16 30 6M14N5M    *  0   0 ATAGCTTCAGC       *
r003 2064 ref 29 17 6H5M       *  0   0 TAGGC             * SA:Z:ref,9,+,5S6M,30,1;
r001   83 ref 37 30 9M         =  7 -39 CAGCGGCAT         * NM:i:1
```

On this basis we can now define the plaintext as being the concatenation of these red blocks. After encrypting them, we divide the cyphertext such that each chunk corresponds to the location of the original.

There are, however, two problems with this approach. First of all, by giving access to the metadata we might in fact grant access to information regarding possible insertions. We know to which block a read belongs. If the combination of the position and the length of the read indicates a discrepancy with the expected end of a block, then there are insertion mutations within that read. One solution for this problem would be to replace the parameter of length of the read, with the length of the alignment. It might nonetheless still be possible to statically infer information about the length of the read through the length of the cypher.

The other problem with this procedure is that the ability to decrypt only portions instead of the whole content is lost. The previously described strategy of providing portions of the one-time pad would work, but it would be extremely hard to use. For example if we want to decrypt the information related to nucleotide 9, we will have to send the one-time pad in order to render the character in green readable:

```
@HD VN:1.5 SO:coordinate
@SQ SN:ref LN:45
r001  163 ref  7 30 8M2I4M1D3M = 37  39 TTAGATAAAGGATACTG *
r002    0 ref  9 30 3S6M1P1I4M *  0   0 AAAAGATAAAGGATA   *
r003    0 ref  9 30 5S6M       *  0   0 GCCTAAGCTAA       * SA:Z:ref,29,-,6H5M,17,0;
r004    0 ref 16 30 6M14N5M    *  0   0 ATAGCTTCAGC       *
```

```
r003 2064 ref 29 17 6H5M       *  0   0 TAGGC            * SA:Z:ref,9,+,5S6M,30,1;
r001   83 ref 37 30 9M         =  7 -39 CAGCGGCAT        * NM:i:1
```

The question is how to perform a read on this partially decrypted data. Due to the fact that the edition steps (information in the sixth column) try to reduce the size of the data, we cannot predict at which character index the information will be stored. Similarly, without knowing if there are any insertions or deletions, we cannot know at which position in the tenth column the nucleotide will be given.

In order to restore the feature of partial block decryption, we would need to undo practically all the attempts to reduce the size, in order to have something where the position of each information can be accurately computed without decrypting everything. This would correspond to the previous model based on FastA/FastQ.

In the end it seems that trying to allow partial decryption on a secured SAM file is not worth the problems it involves. The easier solution seems to reside in coarse-grained rules based on blocks only.

Up to now we have not considered the case where an original read belongs to two or more blocks. The obvious solution to this would be to split it in multiple and shorter reads, each belonging to only one group. In case we would need to reconstruct the original reads in the future, we would have to add extra information to be able to do so.

Based on this decision we can then define a format quite similar to the original SAM. As in the unprotected version, we first need to define the reference in use. We take the first two lines of the original format to do so. Then we iterate over each block we have decided to generate based on our knowledge. However, as mentioned before, the knowledge about which blocks are meaningful will evolve over time, therefore it is far more convenient to define a system which adapts to any decision. Such a solution could consist in adding additional lines which indicate the beginning of a new block. In the proposed version, those lines start with the symbol '#' if the information is provided as plaintext, and '!' if the information is encrypted. If the prediction that there are fewer than 20 000 genes ([42]) is correct, then the number of blocks and therefore lines will not be extremely high compared to the whole document. Each line could also be rather short: just indicating up to which nucleotide index of the reference the current block covers should in theory be sufficient. After this line we would then include either the original reads, or the encrypted original reads.

The end result may then look like this:

```
@HD VN:1.5 SO:coordinate
@SQ SN:ref LN:75
#45
r001  163 ref  7 30 8M2I4M1D3M = 37  39 TTAGATAAAGGATACTG *
```

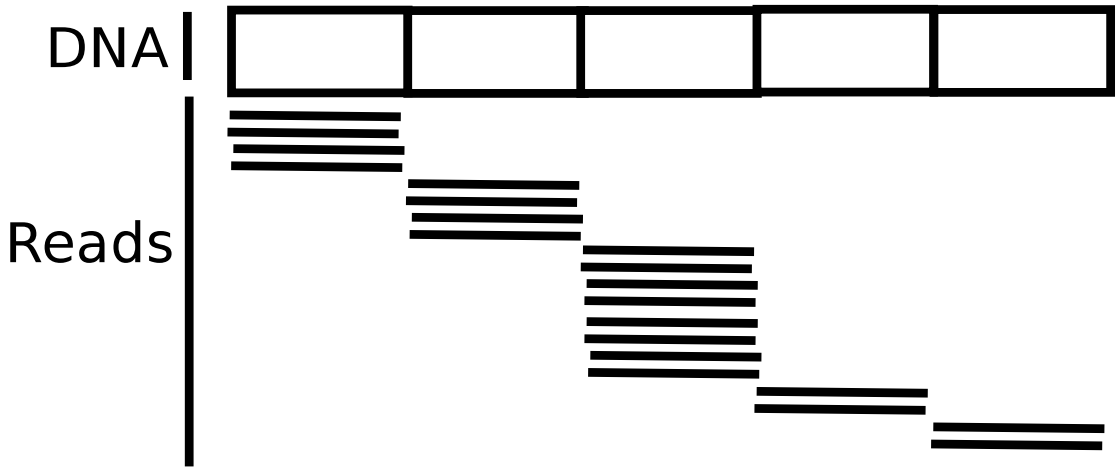


Figure 5.6

```
r002    0 ref  9 30 3S6M1P1I4M *  0   0 AAAAGATAAAGGATA   *
r003    0 ref  9 30 5S6M       *  0   0 GCCTAAGCTAA       * SA:Z:ref,29,-,6H5M,17,0;
r004    0 ref 16 30 6M14N5M    *  0   0 ATAGCTTCAGC       *
r003 2064 ref 29 17 6H5M       *  0   0 TAGGC             * SA:Z:ref,9,+,5S6M,30,1;
r001   83 ref 37 30 9M         =  7 -39 CAGCGGCAT         * NM:i:1
!60
UnReAdAbLe-ThIs_WoUlD_bE_eNcRyPtEd
!45
UnReAdAbLe-ThIs_WoUlD_bE_eNcRyPtEd
```

5.6 Securing the CRAM format

The CRAM structure is quite different from the SAM/BAM format. As shown in Figure 2.1 there are already very clear hierarchical subdivisions. The other difference with SAM is that evaluating the position of each value might be quite hard since CRAM is from the beginning intended to be compressed. One might think that it could be even harder to accept fine-grained decryption in this format than in the case of SAM.

However, using the structure of the file to our advantage it is possible to offer an easy coarse- or medium-grained decryption.

A CRAM file is subdivided in containers which are further divided into slices. We can easily use CRAM's blocks as equivalent to the semantic blocks we have previously used, and the slices would then be subdivisions which can be decrypted separately.

The main element to be encrypted are the Core Data blocks. We can use any of the usual and secure ways to encrypt the compressed reads included in these core blocks. Nevertheless it would be a nice feature to reduce the time needed for communication by limiting as much as possible the data to be sent in order to allow the decryption: in the case of decrypting a whole block, it is easier to send one key which allows the decryption of all slices, rather than a number of keys to decrypt each slice separately.

A possible strategy could be to generate one key k_c for each container. Then we take advantage of the fact that every CRAM block has already an id to generate a key k_{sId} for the slice with that id. The method could be as straightforward as $k_{sId} = E(k_c, Id)$. Then the slice is encrypted using k_{sId}. We would just need to add a new type of flag in the CRAM specification to indicate that the content of the slice is encrypted. If we then want to give a read grant for the slice, we send its key k_{sId}. The downside to this approach is that we are possibly facing a variant of the known plaintext attack.

This is not entirely the case however. In a known plaintext attack, the attacker has access to the cypher and to the plaintext and knows the encryption method at use. Here we have the plaintext (the id of the slice) and the encryption method to create an unknown cypher (k_{sId}) with which to encrypt another somewhat unknown plaintext (the slice) into a known cypher (the data at hand). It might nevertheless be useful to apply another approach, but the change would probably make us store more keys: with this approach we need to store fewer keys, since there are deterministically generated.

The presented method is also advantageous when we want to grant access to an entire container. We would have encrypted it using the k_c key, adding, in this case, an indication in the container header. Each slice would be encrypted according to the previously described approach, leaving only the id of the slice as plaintext. When a patient wants to grant a read access to the whole container, s/he only has to send the key k_c, and the other side will be able to decrypt all the information in the container.

Chapter 6

CONCLUSION

We have seen that the current evolution in genomic studies pushes us to find a solution for storing the generated information. However, this information is extremely private, not just for the person who has been sequenced, but also for the blood-relatives who share part of this information. This is one of the key aspects of DNA which makes it so different from anything else to defend. By its very nature it is impossible to anonymize, it reveals sensitive information which might be even unknown to the patient, and it holds great potential to foster breakthroughs in medicine.

When making DNA records available to either a doctor, a research group, a company offering a service or for another usage yet not devised, we must face a privacy-utility tradeoff problem. This issue is unavoidable, but we have seen that there are strategies to amend it. One idea is to make the individual disappear in a crowd of peers: in order to still have the utility of a DNA record at our disposal we aggregate its information to the one provided by the records of many more individuals. We then just allow to query information over the combination of all data, by resolving statistical tests over the pool. This solution is being applied but as we have seen, special strategies have been devised to break this approach. Even though some doubt the feasibility of such an attack, this fear has spurred different strategies to further defend the privacy of individuals. Some of these strategies are as counter-intuitive as adding noise to the statistical result, which rises the obvious question whether this is truly an improvement to the privacy-utility bargain.

Other approaches rather define access rules. Depending on the person requesting, the intended usage, and the actual query, a policy engine filters the requests and only grants those which are tolerated by the rules defined by the patient. This approach certainly helps to fix fine-grained rules which allow to publish just small portions of the DNA, ensuring thus both privacy and utility.

However, there is still no standard which integrates usage rules within its definition. As MPEG starts a standardization process it is important to show that

such privacy measures can be integrated, even in specifications which are currently in use. We can write rules expressing what is allowed and under which conditions, however this does not protect us against a rogue player who decides not to respect the will expressed. The solution we propose is based on the encryption of entire blocks of content and combines well with a grant request workflow in which the individual can grant the use of just one section by sharing the decryption information for only that particular section.

We have also seen how multiple files of this nature could coexist, each revealing one portion of the DNA but which cannot be recombined, if we can find a suitable metric for the identification potential of a given DNA region.

In order to define the best blocks to encrypt, we will have to balance once again privacy and utility, but in this case maybe also compression potential. We face also other issues with the encryption itself: not that many documents will have utility lifespans extending potentially over the whole life of an individual and his descendants. This means that we will have to face potential changes in the encryption of the file, e.g. upgrading its encryption scheme, an issue we have not considered in this work. It might be very hard to achieve secure later changes: if the document was encrypted with AES for example, and some years later a discovery pushes us to move to a next generation AES2, a potential attacker could keep his AES encrypted version to break it, and there would be no control over this.

The standardization process on a file format for DNA will not be relevant to genetic information only. Other studies in biology will probably take a similar path. The "other omics" bear a potential similar to genomic information: they are supposed to have identification properties and one could analyse these omics in search of marks for certain diseases, which would allow to predict individual health risks.

Glossary

allele Name given to one of the existing variants for a given gene. 7

amino acid The proteins produced by the cells are the concatenation of amino acids. There are a total of 21 different amino acids. 7

BAM Compressed version of SAM. 11

beacon Database for DNA records of multiple individuals: it allows queries concerning statistics over the whole body of data. 18

chromosome Structure made of one DNA molecule, which is only visible during a division process. In a human body, the information is divided in 23 pairs of chromosomes: in each pair, one chromosome is inherited from the father, the other one from the mother. 6

codon In a gene, each group of three nucleotides is called a codon, its role is to encode the next instruction for the molecule to be produced by the cell (i.e. the next amino acid to add). 7

CRAM Compressed binary file format including multiple FASTQ information aligned on a reference genome. 11

Deoxyribonucleic acid Long structure composed of two complementary strands (sequence of nucleotides). 6

DNA Deoxyribonucleic acid. 6

FASTA File format including header and nucleotides sequence as read by the sequencing hardware. 9

FASTQ File format including header and nucleotides sequence as read by the sequencing hardware (including grading of the quality of the read). 9

gene Region of a DNA molecule which encodes a specific protein. 7

MAF Minor allele frequency. 12

meiosis Process by which a cell divides into two sexual cells. The cells resulting of the division receive either DNA information inherited from the father or the mother. 6

Minor Allele Frequency Each allele occurs with a certain frequency; 'Minor Allele Frequency' refers to the frequency of the least common one. 12

mitosis Process by which a cell divides into two cells which have the same information. In this process, the cell generates a copy of each DNA molecule, and the cells resulting of the division receive either the original or the copy. 6

nucleotide Basic coding unit of the DNA. There are four different nucleotides: adenine (A), cytosine (C), guanine (G), thymine(T). 6

SAM Human readable file format including multiple FASTQ information aligned on a reference genome. 10

Single-Nucleotide Polymorphism Mutation involving only one nucleotide (either an insertion, a deletion or a change). 7

SNP Single-Nucleotide Polymorphism. 7

Bibliography

[1] Amy L McGuire, Rebecca Fisher, Paul Cusenza, Kathy Hudson, Mark A Rothstein, Deven McGraw, Stephen Matteson, John Glaser, and Douglas E Henley. Confidentiality, privacy, and security of genetic and genomic test information in electronic health records: points to consider. *Genetics in medicine : official journal of the American College of Medical Genetics*, 10(7):495–9, July 2008.

[2] Genomics and Patient Privacy Conference 2016. `https://med.stanford.edu/gapp/events/GAPPConference2016.html`, 2016. 2016-03-18.

[3] Peter Claes, Harold Hill, and Mark D Shriver. Toward DNA-based facial composites: preliminary results and validation. *Forensic science international. Genetics*, 13:208–16, December 2014.

[4] Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V Pearson, Dietrich A Stephan, Stanley F Nelson, and David W Craig. Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays. *PLoS genetics*, 4(8):e1000167, August 2008.

[5] David Clayton. On inferring presence of an individual in a mixture: a Bayesian approach. *Biostatistics (Oxford, England)*, 11(4):661–73, October 2010.

[6] Kevin B Jacobs, Meredith Yeager, Sholom Wacholder, David Craig, Peter Kraft, David J Hunter, Justin Paschal, Teri A Manolio, Margaret Tucker, Robert N Hoover, Gilles D Thomas, Stephen J Chanock, and Nilanjan Chatterjee. A new statistic and its power to infer membership in a genome-wide association study using genotype frequencies. *Nature genetics*, 41(11):1253–7, November 2009.

[7] Rui Wang, Yong Fuga Li, XiaoFeng Wang, Haixu Tang, and Xiaoyong Zhou. Learning your identity and disease from research papers. In *Proceedings of the 16th ACM conference on Computer and communications security - CCS '09*, page 534, New York, New York, USA, November 2009. ACM Press.

[8] Rosemary Braun, William Rowe, Carl Schaefer, Jinghui Zhang, and Kenneth Buetow. Needles in the haystack: identifying individuals present in pooled genomic data. *PLoS genetics*, 5(10):e1000668, October 2009.

[9] Suyash S. Shringarpure and Carlos D. Bustamante. Privacy Risks from Genomic Data-Sharing Beacons. *The American Journal of Human Genetics*, 97(5):631–46, October 2015.

[10] Cynthia Dwork. *Automata, Languages and Programming: 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*, chapter Differenti, pages 1–12. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[11] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 17–32, San Diego, CA, 2014. USENIX Association.

[12] Daniel Kifer and Ashwin Machanavajjhala. No free lunch in data privacy. In *Proceedings of the 2011 international conference on Management of data - SIGMOD '11*, page 193, New York, New York, USA, 2011. ACM Press.

[13] Raghav Bhaskar, Srivatsan Laxman, Adam Smith, and Abhradeep Thakurta. Discovering frequent patterns in sensitive data. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '10*, page 503, New York, New York, USA, 2010. ACM Press.

[14] Aaron Johnson and Vitaly Shmatikov. Privacy-preserving data exploration in genome-wide association studies. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '13*, page 1079, New York, New York, USA, 2013. ACM Press.

[15] Caroline Uhler, Aleksandra B. Slavkovic, and Stephen E. Fienberg. Privacy-Preserving Data Sharing for Genome-Wide Association Studies. May 2012.

[16] Fei Yu, Stephen E. Fienberg, Aleksandra B. Slavković, and Caroline Uhler. Scalable privacy-preserving data sharing methodology for genome-wide association studies. *Journal of Biomedical Informatics*, 50:133–141, 2014.

[17] Florian Tramèr, Zhicong Huang, Jean-Pierre Hubaux, and Erman Ayday. Differential Privacy with Bounded Priors. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS '15*, pages 1286–1297, New York, New York, USA, October 2015. ACM Press.

[18] Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. The reusable holdout: Preserving validity in adaptive data analysis. *Science*, 349(6248):636–638, August 2015.

[19] Dale R Nyholt, Chang-En Yu, and Peter M Visscher. On Jim Watson's APOE status: genetic information is hard to hide. *European journal of human genetics : EJHG*, 17(2):147–9, March 2009.

[20] Sahel Shariati Samani, Zhicong Huang, Erman Ayday, Mark Elliot, Jacques Fellay, Jean-Pierre Hubaux, and Zoltan Kutalik. Quantifying Genomic Privacy via Inference Attack with High-Order SNV Correlations. In *2015 IEEE Security and Privacy Workshops*, pages 32–40. IEEE, May 2015.

[21] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st annual ACM symposium on Symposium on theory of computing - STOC '09*, page 169, New York, New York, USA, 2009. ACM Press.

[22] Y Chen, B Peng, XF Wang, and H Tang. Large-Scale Privacy-Preserving Mapping of Human Genomic Sequences on Hybrid Clouds. *NDSS*, 2012.

[23] Jung Hee Cheon, Miran Kim, and Kristin Lauter. Homomorphic Computation of Edit Distance. pages 194–212. Springer Berlin Heidelberg, 2015.

[24] Masaya Yasuda, Takeshi Shimoyama, Jun Kogure, Kazuhiro Yokoyama, and Takeshi Koshiba. Secure pattern matching using somewhat homomorphic encryption. *Proceedings of the ACM Conference on Computer and Communications Security*, pages 65–76, November 2013.

[25] Emiliano De Cristofaro, Sky Faber, and Gene Tsudik. Secure genomic testing with size- and position-hiding private substring matching. In *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society - WPES '13*, pages 107–118, New York, New York, USA, 2013. ACM Press.

[26] Ludovic Barman, Mohammed-Taha Elgraini, Jean Louis Raisaro, Jean-Pierre Hubaux, and Erman Ayday. Privacy Threats and Practical Solutions for Genetic Risk Tests. In *2015 IEEE Security and Privacy Workshops*, pages 27–31. IEEE, May 2015.

[27] Wenjie Lu, Yoshiji Yamada, and Jun Sakuma. Efficient Secure Outsourcing of Genome-Wide Association Studies. In *2015 IEEE Security and Privacy Workshops*, pages 3–6. IEEE, May 2015.

[28] David A. Duverle, Shohei Kawasaki, Yoshiji Yamada, Jun Sakuma, and Koji Tsuda. Privacy-Preserving Statistical Analysis by Exact Logistic Regression. In *2015 IEEE Security and Privacy Workshops*, pages 7–16. IEEE, May 2015.

[29] Nathan Dowlin, Ran Gilad-Bachrach, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. Manual for Using Homomorphic Encryption for Bioinformatics. Technical Report MSR-TR-2015-87, 2015.

[30] MPEG requirements "ISO/IEC JTC 1/SC 29/WG 11 N16134 - Requirements on Genome Compression and Storage". 2016.

[31] S. Llorente J. Delgado. ISO/IEC JTC1/SC29/WG11 MPEG2016/m37802 - Follow up on application scenarios for privacy and security requirements on genome usage, compression, transmission and storage. 2016.

[32] Global Alliance for Genomics and Health. Standards and implementation practices for protecting the privacy and security of shared genomic and clinical data.

[33] Emiliano De Cristofaro. An Exploratory Ethnographic Study of Issues and Concerns with Whole Genome Sequencing. June 2013.

[34] Mathias Humbert, Erman Ayday, Jean-Pierre Hubaux, and Amalio Telenti. Reconciling Utility with Privacy in Genomics. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society - WPES '14*, pages 11–20, New York, New York, USA, 2014. ACM Press.

[35] Michael T. Goodrich. The Mastermind Attack on Genomic Data. In *2009 30th IEEE Symposium on Security and Privacy*, pages 204–218. IEEE, May 2009.

[36] Richman Wee, Mark Henaghan, and Ingrid Winship. Dynamic consent in the digital age of biology: online initiatives and regulatory considerations. *Journal of primary health care*, 5(4):341–7, December 2013.

[37] William G Dixon, Karen Spencer, Hawys Williams, Caroline Sanders, David Lund, Edgar A Whitley, and Jane Kaye. A dynamic model of patient consent to sharing of medical record data. *BMJ (Clinical research ed.)*, 348(feb05_5):g1294, January 2014.

[38] Jane Kaye, Edgar A Whitley, David Lund, Michael Morrison, Harriet Teare, and Karen Melham. Dynamic consent: a patient interface for twenty-first century research networks. *European journal of human genetics : EJHG*, 23(2):141–6, February 2015.

[39] Jane Kaye, Liam Curren, Nick Anderson, Kelly Edwards, Stephanie M Fullerton, Nadja Kanellopoulou, David Lund, Daniel G MacArthur, Deborah Mascalzoni, James Shepherd, Patrick L Taylor, Sharon F Terry, and Stefan F

Winter. From patients to partners: participant-centric initiatives in biomedical research. *Nature reviews. Genetics*, 13(5):371–6, May 2012.

[40] Deborah Mascalzoni, Andrew Hicks, and Peter P Pramstaller. Consenting in Population Genomics as an Open Communication Process. *Studies in Ethics, Law, and Technology*, 3(1), January 2009.

[41] Miguel Freitas. twister - a P2P microblogging platform. `http://arxiv.org/abs/1312.7152`, December 2013.

[42] Iakes Ezkurdia, David Juan, Jose Manuel Rodriguez, Adam Frankish, Mark Diekhans, Jennifer Harrow, Jesus Vazquez, Alfonso Valencia, and Michael L. Tress. The shrinking human protein coding complement: are there now fewer than 20,000 genes? December 2013.

[43] Stephen E. Fienberg, Aleksandra Slavkovic, and Caroline Uhler. Privacy Preserving GWAS Data Sharing. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pages 628–635. IEEE, December 2011.

[44] Markus Hsi-Yang Fritz, Rasko Leinonen, Guy Cochrane, and Ewan Birney. Efficient storage of high throughput DNA sequencing data using reference-based compression. *Genome research*, 21(5):734–40, May 2011.

[45] David A Wheeler, Maithreyan Srinivasan, Michael Egholm, Yufeng Shen, Lei Chen, Amy McGuire, Wen He, Yi-Ju Chen, Vinod Makhijani, G Thomas Roth, Xavier Gomes, Karrie Tartaro, Faheem Niazi, Cynthia L Turcotte, Gerard P Irzyk, James R Lupski, Craig Chinault, Xing-zhi Song, Yue Liu, Ye Yuan, Lynne Nazareth, Xiang Qin, Donna M Muzny, Marcel Margulies, George M Weinstock, Richard A Gibbs, and Jonathan M Rothberg. The complete genome of an individual by massively parallel DNA sequencing. *Nature*, 452(7189):872–6, April 2008.

[46] Yaniv Erlich and Arvind Narayanan. Routes for breaching and protecting genetic privacy. *Nature reviews. Genetics*, 15(6):409–21, June 2014.

[47] Murat Kantarcioglu, Wei Jiang, Ying Liu, and Bradley Malin. A cryptographic approach to securely share and query genomic sequences. *IEEE transactions on information technology in biomedicine : a publication of the IEEE Engineering in Medicine and Biology Society*, 12(5):606–17, September 2008.

[48] Mathias Humbert, Erman Ayday, Jean-Pierre Hubaux, and Amalio Telenti. Addressing the concerns of the lacks family. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security - CCS*

'13, pages 1141–1152, New York, New York, USA, November 2013. ACM Press.

[49] Miran Kim and Kristin Lauter. Private Genome Analysis through Homomorphic Encryption. Cryptology ePrint Archive, Report 2015/965, 2015.

[50] L.Beskow L. Wolf, E. Fuse. If we don't own our genes, what protects study subjects in genetic research? `http://theconversation.com/if-we-dont-own-our-genes-what-protects-study-subjects-in-genetic-research-56003`. 2016-04-03.

[51] Oded Goldreich and Rafail Ostrovsky. Software protection and simulation on oblivious RAMs. *Journal of the ACM*, 43(3):431–473, May 1996.

[52] Mark Phillips, Bartha M. Knoppers, and Yann Joly. Seeking a "Race to the Top" in Genomic Cloud Privacy? In *2015 IEEE Security and Privacy Workshops*, pages 65–69. IEEE, May 2015.

[53] Keisuke Tanaka and Yuji Suga, editors. *Advances in Information and Computer Security*, volume 9241 of *Lecture Notes in Computer Science*. Springer International Publishing, Cham, 2015.

[54] Xiao Shaun Wang, Yan Huang, Yongan Zhao, Haixu Tang, XiaoFeng Wang, and Diyue Bu. Efficient Genome-Wide, Privacy-Preserving Similar Patient Query based on Private Edit Distance. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS '15*, pages 492–503, New York, New York, USA, October 2015. ACM Press.

[55] Mete Akgün, A Osman Bayrak, Bugra Ozer, and M Şamil Sağıroğlu. Privacy preserving processing of genomic data: A survey. *Journal of biomedical informatics*, 56:103–11, August 2015.

[56] Muhammad Naveed, Erman Ayday, Ellen W Clayton, Jacques Fellay, Carl A Gunter, Jean-Pierre Hubaux, Bradley A Malin, and Xiaofeng Wang. Privacy in the Genomic Era. *ACM computing surveys*, 48(1):6, September 2015.

[57] Nikolaos Karvelas, Andreas Peter, Stefan Katzenbeisser, Erik Tews, and Kay Hamacher. Privacy-Preserving Whole Genome Sequence Processing through Proxy-Aided ORAM. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society - WPES '14*, pages 1–10, New York, New York, USA, November 2014. ACM Press.

[58] Isabel Wagner. Genomic Privacy Metrics: A Systematic Comparison. In *2015 IEEE Security and Privacy Workshops*, pages 50–59. IEEE, May 2015.

[59] Sean Simmons and Bonnie Berger. One Size Doesn't Fit All: Measuring Individual Privacy in Aggregate Genomic Data. In *2015 IEEE Security and Privacy Workshops*, pages 41–49. IEEE, May 2015.

[60] Kristin Lauter, Adriana Lopez-Alt, and Michael Naehrig. Private Computation on Encrypted Genomic Data. Technical Report MSR-TR-2014-93, 2014.

[61] Liina Kamm, Dan Bogdanov, Sven Laur, and Jaak Vilo. A new way to protect privacy in large-scale genome-wide association studies. *Bioinformatics (Oxford, England)*, 29(7):886–93, April 2013.

[62] Barbara Prainsack and Alena Buyx. A solidarity-based approach to the governance of research biobanks. *Medical law review*, 21(1):71–91, January 2013.

[63] John P A Ioannidis. Why most published research findings are false. *PLoS medicine*, 2(8):e124, August 2005.

[64] Yaniv Erlich, James B Williams, David Glazer, Kenneth Yocum, Nita Farahany, Maynard Olson, Arvind Narayanan, Lincoln D Stein, Jan A Witkowski, and Robert C Kain. Redefining genomic privacy: trust and empowerment. *PLoS biology*, 12(11):e1001983, November 2014.

[65] Sriram Sankararaman, Guillaume Obozinski, Michael I Jordan, and Eran Halperin. Genomic privacy and limits of individual detection in a pool. *Nature genetics*, 41(9):965–7, September 2009.

[66] W. Bonyhon and Bruce A. Gene testing framework ignores privacy and security concerns. `http://theconversation.com/gene-testing-framework-ignores-privacy-and-security-concerns-12650`.

[67] A. Cavoukian. Privacy by Design The 7 Foundational Principles. `https://www.iab.org/wp-content/IAB-uploads/2011/03/fred_carter.pdf`.

[68] Privacy and protection in the genomic era. *Nature medicine*, 19(9):1073, September 2013.

[69] Erman Ayday, Emiliano De Cristofaro, Jean-Pierre Hubaux, and Gene Tsudik. Whole Genome Sequencing: Revolutionary Medicine or Privacy Nightmare? *Computer*, 48(2):58–66, February 2015.